

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

BEE-STEWARD

Comments to changes in the code

BEE-STEWARD merges the bumblebee population model *Bumble-BEEHAVE* with the landscape defining features of BEESCOUT and while offering a simplified, user-friendly interface. The full documentation of these two underlying models is provided on the BEEHAVE website as well as in the supplementary material of their publications.

Here, we describe the major changes and additions of BEE-STEWARD in comparison with the underlying models. These changes affect the setup processes, the way maps/landscapes are defined etc. No significant changes have been made to the actual model processes, i.e. to the procedures called by "Go" to run the Bumble-BEEHAVE model. The the agent-based scouting module of BEESCOUT has *not* been added to BEE-STEWARD. Furthermore, some parameters may have been renamed and more output variables have been defined.

REFERENCES:

Becher, M. A., Grimm, V., Knapp, J., Horn, J., Twiston-Davies, G., & Osborne, J. L. (2016). BEESCOUT: A model of bee scouting behaviour and a software tool for characterizing nectar/pollen landscapes for BEEHAVE. *Ecological Modelling*, 340, 126-133.

Becher, M. A., Twiston-Davies, G., Penny, T. D., Goulson, D., Rotheray, E. L., & Osborne, J. L. (2018). *Bumble-BEEHAVE*: A systems model for exploring multifactorial causes of bumblebee decline at individual, colony, population and community level. *Journal of Applied Ecology*, 55(6), 2790-2801.

New breeds

Four new breeds were added to BEE-STEWARD: *habitats*, *buttons*, *buttonLabels* and *brushSigns*.

Habitats

The new breed *habitats* save the information of a habitat type, i.e. a list of plant species and densities, the colour range associated with the habitat etc.

```
habitats-own
[
  flowerspecieslist
  habitatColourID
  colourRangeMin
  colourRangeMax
  habitatType
  habitatSwitchedOn?
]
```

Buttons & ButtonLabels

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

The new breeds *buttons* and *buttonLabels* are used to display a virtual panel, where the user can see and modify the parameterisation of the model. They don't have any breed's "-own" variables defined.

BrushSigns

The new breed *brushSigns* defines the shape of the brush, when a user is modifying the map by drawing onto it.

Setup Procedures

Setup

The procedure *Setup* calls two procedures, which makes it possible to update parameters without using the *clear-all* commands, which sets all variables to their initial value.

```
to Setup
  ClearProc
  SetupWithoutClearingProc
end
```

ClearProc

ClearProc removes all agents, all grid cells are set to their default initial values, all global variables, including the time steps, are set to 0, plots and other output on the interface is cleared.

```
to ClearProc
  clear-all
  stop-inspecting-dead-agents
  reset-ticks
end
```

SetupWithoutClearingProc

SetupWithoutClearingProc is similar to the *Bumble-BEEHAVE* procedure *Setup*, except that clearing of variables has been moved to the procedure *ClearProc*. Additionally, the procedures *PanelSettingProc* and *ReadAllParametersProc* are called.

```
to SetupWithoutClearingProc
  if RAND_SEED != 0 [ random-seed RAND_SEED ]
  ;; OR ALTERNATIVELY:
  ;; if RAND_SEED != 0 [ random-seed RAND_SEED + 10000 * behaviorspace-run-number ]
  if (MyMap = "_SYSTEM_Example_Farm.png"
    and MyParametersFile = "_SYSTEM_Example_Farm_Parameters.csv")
    [ set ProjectsOwnParameterFile? false ] ; to make sure that under default setting,
    ; the default (System) parameter file is used
  if ProjectsOwnParameterFile? = true
  [
    set MyParametersFile (word remove "_SYSTEM_" MyMap) ; the parameter file of this
    ; project has the same name as the map used, but make sure, system files
    ; cannot be overwritten
    set MyParametersFile (word remove ".png" MyParametersFile) ; MyMap is an image file but
    ; MyParametersFile is not - delete extension!
  ]
end
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

    set MyParametersFile (word MyParametersFile "_Parameters.csv") ; identifier for
                          ; parameter file and correct extension
]
PanelSettingProc
ReadAllParametersProc
ParametersProc
CreateFoodsourcesProc
CreateSpeciesProc
CreateBadgersProc
CreateInitialQueensProc
UpdateMorning_Proc
CreateSignsProc
OutputDailyProc
if ShowGrid? = true
[
    ask patches with [ remainder pxcor round (Gridsize * Scaling_NLpatches/m) = 0 ]
    [ set pcolor black ]
    ask patches with [ remainder pycor round (Gridsize * Scaling_NLpatches/m) = 0 ]
    [ set pcolor black ]
    ask patch 290 5 [ set plabel-color black set plabel word Gridsize " m" ]
]
if (MyMap = "_SYSTEM_Example_Farm.png"
    and MyParametersFile = "_SYSTEM_Example_Farm_Parameters.csv")
[ set ProjectsOwnParameterFile? true ] ; to make sure that under default setting,
; the default (System) parameter file is used
if count foodsources with [patchtype = "undefined"] > 0
[ AssertionProc "Undefined habitat patch!" ]
end

```

PanelSettingProc

Sets up the "Control Panel" on the interface. Depending on the selected option of *Panel*, the buttons 1 - 7 run different commands, specified by their associated monitors.

```

to PanelSettingProc
if Panel = "Modify Maps"
[
    let notBrushShape "Circular"
    if CircularBrush? = true [ set notBrushShape "Square" ]
    set Button1Monitor (word "Choose Colour (" SetColour ")")
    set Button2Monitor (word "Set Brush Size (" BrushSize ")
                          (ca. " precision BrushArea_ha 1 "ha)")
    set Button3Monitor (word "Switch Brush Shape to " notBrushShape)
    set Button4Monitor "Draw on Map"
    set Button5Monitor "Replace Colours"
    set Button6Monitor "Clear whole Map"
    set Button7Monitor "Update current Map"
]

if Panel = "Stewardship Options"
[
    set Button1Monitor (word "Select Stewardship Option (" StewardshipOption ")")
    set Button2Monitor "Select Field"
    set Button3Monitor "Apply Stewardship"
    set Button4Monitor "Show Stewardship Areas"
    set Button5Monitor "Define Crop Rotation"
    set Button6Monitor "Unselect all Fields"
    set Button7Monitor "Generate My Report!"
]

if Panel = "Maps and Settings"
[
    set Button1Monitor "Load Existing Map"
    set Button2Monitor "Create Map from Scan or load GIS text file"
    set Button3Monitor "Set Parameter Values"
    set Button4Monitor "Load Setting"
    set Button5Monitor "Save Setting"
    set Button6Monitor "Show or hide Scale Bar"
    set Button7Monitor "Initial Queens"
]

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

if Panel = "Display Options"
[
  set Button1Monitor "Default view"
  set Button2Monitor "Show Nectar Visits"
  set Button3Monitor "Show Pollen Visits"
  set Button4Monitor "Identify!"
  set Button5Monitor "Show Input Files"
  set Button6Monitor "Show Parameter Values"
  set Button7Monitor "More Display Options"
]

if Panel = "Advanced Input Options"
[
  set Button1Monitor (word "Set Random Seed (" RAND_SEED ")")
  set Button2Monitor "Advanced Setup Options"
  set Button3Monitor "Add a Background Image"
  set Button4Monitor ""
  set Button5Monitor ""
  set Button6Monitor ""
  set Button7Monitor "VERSION TEST"
]

if Panel = "My Own Maps"
[
  set Button1Monitor (word "Load Map 1 (" MySavedMap1 ")")
  set Button2Monitor (word "Load Map 2 (" MySavedMap2 ")")
  set Button3Monitor (word "Load Map 3 (" MySavedMap3 ")")
  set Button4Monitor (word "Load Map 4 (" MySavedMap4 ")")
  set Button5Monitor (word "Load Map 5 (" MySavedMap5 ")")
  set Button6Monitor (word "Load Map 6 (" MySavedMap6 ")")
  set Button7Monitor "Delete one of My Maps"
]

if Panel = ""
[
  set Button1Monitor ""
  set Button2Monitor ""
  set Button3Monitor ""
  set Button4Monitor ""
  set Button5Monitor ""
  set Button6Monitor ""
  set Button7Monitor ""
] end

```

ReadAllParametersProc

Reads in the parameter file, that is associated with the current map. If this file doesn't exist, the default parameter file is loaded.

to ReadAllParametersProc

```

;; if ProjectsOwnParameterFile? = true:
;; MyParametersFile is set to the project's default parameter file name
;; ("projectname_Parameters.csv").
;; If this file exists, it is loaded, if it does not exist, it is created
;; from "_SYSTEM_Parameters.csv"
;;
;; if ProjectsOwnParameterFile? = false:
;; A) if MyParametersFile exists, it is loaded as parameter file
;; B) otherwise, MyParametersFile is set to "_SYSTEM_Parameters.csv" (and loaded,
;; without being saved under a new name)
;;

let newParameterfileNeedsToBeCreated? false
let parametersDataCSV []
ifelse file-exists? MyParametersFile
  [ set parametersDataCSV csv:from-file MyParametersFile ] ; a csv input file is loaded
                                     ; and saved

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
[
  ifelse file-exists? "_SYSTEM_Parameters.csv"
  [
    output-print "MyParametersFile was created from '_SYSTEM_Parameters.csv'"
    set parametersDataCSV csv:from-file "_SYSTEM_Parameters.csv"
    set MyParametersFile "_SYSTEM_Parameters.csv"

    if ProjectsOwnParameterFile? = true [ set newParameterfileNeedsToBeCreated? true ]
  ]
  [
    output-print "Neither the specified MyParametersFile nor '_SYSTEM_Parameters.csv' do
      exist in this folder!"
    user-message "Can't find input file '_SYSTEM_Parameters.csv'"
  ]
]
let header item 0 parametersDataCSV ; first line of the input file is the header
set AllParametersList header
let parameterDataList item 1 parametersDataCSV

foreach header
[
  let nextCommand ""
  let newEntry (word item (position (word ?) header) parameterDataList)
  if member? "\"\"" newEntry and ? != "MyFarmlandPatches" ; if margings are added
  ;; doubled double-quotes can occur. They need to be replaced by single double-quotes.
  ;; Exception for MyFarmlandPatches, which actually can be ""!
  [
    set newEntry remove "\"\" newEntry
    set newEntry (word "\"\" newEntry "\"")
  ]
  set nextCommand (word "set " ? " " newEntry)
  run nextCommand
]
if newParameterfileNeedsToBeCreated? = true [ SaveLoadSettingsProc "Save!" "" ]
end
```

SetupBehaviorSpace

This procedure replaces *Setup*, if the model is run via NetLogo's BehaviorSpace. As parameter values are saved in an input file, these parameters cannot be defined in the usual way in BehaviourSpace, as they would be overwritten during the setup, hence, they need to be saved in parameter files. This procedure creates a unique parameter file for each automated model run, sets up the model and then deletes this temporary parameter file again.

to SetupBehaviorSpace ; only called when BehaviorSpace is used!

```
  ; 1.) When running a BehaviorSpace experiment, several cores might try to access and modify
  ; the parameter file at the same time, which causes errors. To avoid this, each run creates its
  ; own (temporary) parameter file, which is deleted at the end of the setup.
```

```
  ; 2.) In contrast to Bumble-BEEHAVE, most parameters of the model are no longer defined via
  ; input options on the GUI but read in. When using BehaviorSpace, this causes problems,
  ; because: either parameters are read in during Setup - which overwrites the values set in
  ; BehaviorSpace or this isn't done, which sets all parameters (except those defined in
  ; BehaviorSpace) to 0.
```

```
  ; This procedure solves this problem by (locally) saving the parameters (and their values)
  ; set in BehaviorSpace and then - after reading in the default values of all parameters,
  ; resetting the BehaviorSpace parameters to the previously saved values.
```

```
  ; This is possible, as "clear-all" doesn't re-set local variables.
```

```
  ; (However, the "run" command only works on global variables, so
```

```
  ; "BehavSpaceParameterValuesList" and "BehavSpaceCurrentValue" both have to be defined as
```

```
  ; global variables. "behavSpaceParameterVALUESListMemo" saves "BehavSpaceParameterValuesList"
```

```
  ; as a local variable (i.e. after Setup/ clear-all, BehavSpaceParameterValuesList will be 0
```

```
  ; but behavSpaceParameterVALUESListMemo will still have the correct values)
```

```
  ; Also note: BehaviourSpaceParameters is a string (as lists cannot be defined in
```

```
  ; BehaviorSpace) and needs to be translated into a list first!)
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

let temporaryParameterFilename (word "____ParameterFileTemporary_"
    behaviorspace-run-number ".csv") ; name of the temporary parameter file
set ProjectsOwnParameterFile? false ; this needs to be false, otherwise, the parameter file
    ; name is automatically derived from the name of the map, will be used
set BehavSpaceParameterValuesList [] ; this variable is a list!

; first, save BehaviourSpaceParameters as a list:
let behaviourSpaceParametersListMemo []
let continue? true
if member? " " BehaviourSpaceParameters = false [ set continue? false ] ; i.e. if only
    ; one parameter (or none at all) is listed in BehaviourSpaceParameters
while [ continue? = true ]
[
    let nextBlankPosition position " " BehaviourSpaceParameters
    let nextItem substring BehaviourSpaceParameters 0 nextBlankPosition ; the first parameter
        ; in the string (incl. first character but not the blank)
    set BehaviourSpaceParameters substring BehaviourSpaceParameters
        (nextBlankPosition + 1) length BehaviourSpaceParameters ; removes first
        ; parameter in the list as well as the first blank
    set behaviourSpaceParametersListMemo lput nextItem behaviourSpaceParametersListMemo
    if member? " " BehaviourSpaceParameters = false [ set continue? false ]
]

if BehaviourSpaceParameters != ""
[
    set behaviourSpaceParametersListMemo lput BehaviourSpaceParameters
        behaviourSpaceParametersListMemo ; BehaviourSpaceParameters here represents the
        ; last parameter, which needs to be added to the list
]

; save the current values of the parameters set in BehaviorSpace in the new
; list BehavSpaceParameterValuesList
if length BehaviourSpaceParameters > 0
[
    foreach behaviourSpaceParametersListMemo
    [
        let nextCommand (word "set BehavSpaceCurrentValue " ? ) ; extra command here (replacing
            ; BehavSpaceCurrentValue in the next command by "?" creates an error)
        run nextCommand
        set nextCommand (word "set BehavSpaceParameterValuesList lput " BehavSpaceCurrentValue "
            BehavSpaceParameterValuesList") ; "run" cannot access local variable!
        run nextCommand
    ]
]

let behavSpaceParameterVALUESListMemo BehavSpaceParameterValuesList ; the global
    ; BehavSpaceParameterValuesList is saved as local variable to survive clear-all!
let beeSpeciesInitialQueensListAsStringMEMO BeeSpeciesInitialQueensListAsString
let cropRotationListAsStringMEMO CropRotationListAsString

; set the parameters file to the default, "_SYSTEM_Parameters.csv" and read it in!
set MyParametersFile "_SYSTEM_Parameters.csv"
ReadAllParametersProc ; the new parameter settings are loaded
Setup
set ProjectsOwnParameterFile? false ; ProjectsOwnParameterFile? may be set true
    ; in Setup (for default map and parameter file) - but it MUST be FALSE in
    ; this procedure!
; all parameters are set to their default values now - those defined in BehaviorSpace
; need to be re-set:
set StopExtinct? false ; make sure each run completes all time steps!
    ; (to set it true, list it in BehaviorSpace!)
if behaviourSpaceParametersListMemo != [] ; i.e. only if there is actually a
    ; parameter defined that needs to be changed
[
    let i 0
    foreach behaviourSpaceParametersListMemo
    [
        let newValue item i behavSpaceParameterVALUESListMemo
        let nextCommand (word "set " ? " " newValue)
        run nextCommand
        set i i + 1
    ]
]
set BeeSpeciesInitialQueensListAsString beeSpeciesInitialQueensListAsStringMEMO
set CropRotationListAsString cropRotationListAsStringMEMO

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

if CropRotationListAsString = 0 [set CropRotationListAsString "" ] ; if using more
    ; than 1 core, CropRotationListAsString is 0 (unless it is listed in BehavSpace),
    ; causing an error ("MEMBER? expected input to be a string or list or agentset but
    ; got the number 0 instead.")
set MyParametersFile temporaryParameterFilename

; all parameters are now set to their correct value. Save these in the parameter file
; and continue with Setup (and run):
SaveLoadSettingsProc "Save!" ""
Setup ; now setup again with the correct parameterisation!

if file-exists? temporaryParameterFilename [ file-delete temporaryParameterFilename ] ; and
    ; finally delete the temporary parameter file!
end

```

Panel Buttons

PanelButtonProc

This procedure is called, whenever one of the seven panel buttons is clicked on. The local variable *myButtonCommand* saves the command of the monitor, that is linked to the activated button (e.g. "Button 1" is linked to the monitor "Button 1", which shows a text defined by *Button1Monitor* (e.g. "Select Field", hence *myButtonCommand* would be set to "Select Field"). The procedure then either runs the code defined by *myButtonCommand* or calls another procedure, that is linked to the command of the button.

```

to PanelButtonProc [ buttonNumber ]
let myButtonCommand ""
if buttonNumber = 1 [ set myButtonCommand Button1Monitor ]
if buttonNumber = 2 [ set myButtonCommand Button2Monitor ]
if buttonNumber = 3 [ set myButtonCommand Button3Monitor ]
if buttonNumber = 4 [ set myButtonCommand Button4Monitor ]
if buttonNumber = 5 [ set myButtonCommand Button5Monitor ]
if buttonNumber = 6 [ set myButtonCommand Button6Monitor ]
if buttonNumber = 7 [ set myButtonCommand Button7Monitor ]

;; PANEL: "STEWARDSHIP OPTIONS"
if myButtonCommand = "Select Field" [ ButtonSelectFieldProc ]
if member? "Select Stewardship Option" myButtonCommand
[
    set StewardshipOption user-one-of "Select a Stewardship Option"
    [ "legume" "margin" "plot" ]
    set Button3Monitor (word "Select Stewardship Option (" StewardshipOption ")")
]

if myButtonCommand = "Unselect all Fields"
[
    foreach SelectedFieldsList [ ask foodsources with [ masterpatchID = ? ]
        [ set shape "circle" ] ]
    set SelectedFieldsList []
]
if myButtonCommand = "Apply Stewardship" [ ButtonStewardshipOptionsProc ]
if myButtonCommand = "Show Stewardship Areas" [ StewardshipAreasProc ]
if myButtonCommand = "Generate My Report!" [ ButtonGenerateOutputProc ]
if myButtonCommand = "Define Crop Rotation" [ CropRotationSelectFilesProc ]

;; PANEL: "MODIFY MAPS"
if myButtonCommand = "Clear whole Map"
[
    ask turtles [ hide-turtle ]
    ask patches [ set pcolor grey ]
]
if member? "Choose Colour" myButtonCommand [ ButtonChooseColourProc ]
if myButtonCommand = "Draw on Map" [ DrawProc ]
if member? "Set Brush Size" myButtonCommand [ ButtonBrushSizeProc ]
if myButtonCommand = "Update current Map"
[
    ;; ButtonDefineFarmareaProc ;; this procedure is currently not in use
    ButtonUseCurrentMapProc TRUE
]

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

]
if myButtonCommand = "Switch Brush Shape to Circular"
[
    set CircularBrush? true
    set Button5Monitor "Switch Brush Shape to Square"
    DrawProc
]
if myButtonCommand = "Switch Brush Shape to Square"
[
    set CircularBrush? false
    set Button5Monitor "Switch Brush Shape to Circular"
    DrawProc
]
if myButtonCommand = "Replace Colours" [ ButtonReplaceColoursProc ]

;; PANEL: "MAPS & SETTINGS"
if myButtonCommand = "Load Existing Map" [ ButtonLoadExistingMapProc ]
if myButtonCommand = "Save Setting" [ SaveLoadSettingsProc "Save!" "" ]
if myButtonCommand = "Load Setting"
[
    SaveLoadSettingsProc "Load!" ""
    SaveLoadSettingsProc "Save!" ""
]
if myButtonCommand = "Create Map from Scan or load GIS text file"
[ BS_ImportNewMapProc TRUE ] ; true: user input required!
if myButtonCommand = "Show or hide Scale Bar" [ ButtonScaleBarProc ]
if myButtonCommand = "Initial Queens" [ ButtonInitialQueensProc ]

;; PANEL: "DISPLAY OPTIONS"
if myButtonCommand = "Default view" [ ButtonDisplayProc "defaultView" ]
if myButtonCommand = "Show Nectar Visits" and max [cumulNectarVisits] of foodsources > 0
[ ButtonDisplayProc "nectarVisits" ]
if myButtonCommand = "Show Pollen Visits" and max [cumulPollenVisits] of foodsources > 0
[ ButtonDisplayProc "pollenVisits" ]
if myButtonCommand = "Identify!" [ ButtonIdentifyProc ]
if myButtonCommand = "Show Parameter Values"
[
    foreach AllParametersList
    [
        let parameter remove " " ? ;; some parameters have a blank added to the end of their
        ;; name, which is removed here
        let command (word "set GenericRunCommandValue " parameter)
        run command
        output-print (word parameter ": " GenericRunCommandValue)
    ]
]

if myButtonCommand = "Show Input Files"
[
    output-type "Input file: " output-print FoodsourcesFile
    output-type "Parameter file: " output-print MyParametersFile
    output-type "Habitat file: " output-print HabitatsFile
    output-type "Flower species file: " output-print FlowerspeciesFile
    output-type "Bumblebee species file: " output-print BeespeciesFile
    output-type "TextMap file: " output-print TextMap
]

if myButtonCommand = "More Display Options" [ ButtonDisplayButtonsProc ]

;; PANEL: "ADVANCED INPUT OPTIONS"
if myButtonCommand = "Advanced Setup Options" [ ButtonAdvancedSetupOptionsProc ]
if myButtonCommand = "Add a Background Image" [ ButtonBackgroundImageProc ]
if myButtonCommand = "Set Parameter Values" [ ParametersSetManuallyProc ]
if member? "Set Random Seed " myButtonCommand
[
    set RAND_SEED read-from-string user-input "New value for RAND_SEED (if 0: random-seed
                                                is not set): "
    set Button1Monitor (word "Set Random Seed (" RAND_SEED ")")
]
if myButtonCommand = "VERSION TEST" [ VersionTestProc ]
if myButtonCommand = "" [ ]

;; PANEL: "MY OWN MAPS"
if myButtonCommand = "Delete one of My Maps"
[

```


Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
let deleteCommand user-one-of "Choose which of your saved maps should be deleted: "  
  [ "Delete My Saved Map 1" "Delete My Saved Map 2" "Delete My Saved Map 3"  
    "Delete My Saved Map 4" "Delete My Saved Map 5" "Delete My Saved Map 6"  
    "Delete ALL My Saved Maps!" ]  
if deleteCommand = "Delete My Saved Map 1" [ set MySavedMap1 "" ]  
if deleteCommand = "Delete My Saved Map 2" [ set MySavedMap2 "" ]  
if deleteCommand = "Delete My Saved Map 3" [ set MySavedMap3 "" ]  
if deleteCommand = "Delete My Saved Map 4" [ set MySavedMap4 "" ]  
if deleteCommand = "Delete My Saved Map 5" [ set MySavedMap5 "" ]  
if deleteCommand = "Delete My Saved Map 6" [ set MySavedMap6 "" ]  
if deleteCommand = "Delete ALL My Saved Maps!"  
  [  
    set MySavedMap1 ""  
    set MySavedMap2 ""  
    set MySavedMap3 ""  
    set MySavedMap4 ""  
    set MySavedMap5 ""  
    set MySavedMap6 ""  
  ]  
  PanelSettingProc  
]  
if member? "Load Map 1" myButtonCommand  
  [  
    ifelse MySavedMap1 = ""  
      [ set MySavedMap1 MyMap ]  
      [ set MyMap MySavedMap1 ]  
    Setup  
  ]  
if member? "Load Map 2" myButtonCommand  
  [  
    ifelse MySavedMap2 = ""  
      [ set MySavedMap2 MyMap ]  
      [ set MyMap MySavedMap2 ]  
    Setup  
  ]  
if member? "Load Map 3" myButtonCommand  
  [  
    ifelse MySavedMap3 = ""  
      [ set MySavedMap3 MyMap ]  
      [ set MyMap MySavedMap3 ]  
    Setup  
  ]  
if member? "Load Map 4" myButtonCommand  
  [  
    ifelse MySavedMap4 = ""  
      [ set MySavedMap4 MyMap ]  
      [ set MyMap MySavedMap4 ]  
    Setup  
  ]  
if member? "Load Map 5" myButtonCommand  
  [  
    ifelse MySavedMap5 = ""  
      [ set MySavedMap5 MyMap ]  
      [ set MyMap MySavedMap5 ]  
    Setup  
  ]  
if member? "Load Map 6" myButtonCommand  
  [  
    ifelse MySavedMap6 = ""  
      [ set MySavedMap6 MyMap ]  
      [ set MyMap MySavedMap6 ]  
    Setup  
  ]  
]  
end
```

Panel "Maps and Settings"

Button 1 "Load Existing Map"

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

This button calls the procedure *ButtonLoadExistingMapProc*. It opens a user dialogue, asking to select a an image file of a previously created BEE-STEWARD map. This map, including the connected foodsources and parameter file, is then opened.

ButtonLoadExistingMapProc

```
to ButtonLoadExistingMapProc
  ; only called by button!
  user-message "Choose one of the maps you have created (image '.png' file)"
  let myChosenMap user-file ; reports the complete path - however, only
                           ; the filename is required!
                           ; e.g. C:\BEE_STEWARD\Example_Farm.png

  if myChosenMap != false
  [
    set myChosenMap reverse myChosenMap ;; as position reports only the first
    ;; (and not the last) occurrence, string is reversed
    ;; e.g. gnp.mraF_elpmaxE\DRAWETS_EEB\C
    let cutHere position "\\\" myChosenMap ; this is the file name + extension, but reversed
    set myChosenMap substring myChosenMap 4 cutHere ; reversed file name without extension
    ; e.g. mraF_elpmaxE
    set myChosenMap reverse myChosenMap ; this is the actual filename
    set myChosenMap remove "_Foodsources" myChosenMap ; in case the user has chosen the
    ; foodsource text, instead of the map
    set myChosenMap remove "_Parameters" myChosenMap ; in case a parameter (csv) file
    ; was chosen
    set MyParametersFile (word myChosenMap "_Parameters.csv") ; correct extensions are added
    set MyMap (word myChosenMap ".png")
    Setup
  ]
end
```

Button 2 "Create Map from Scan or load GIS text file"

Calls the procedure *BS_ImportNewMapProc*. This procedure has a Boolean input variable *userInput?* which, if set true, will open user dialogues asking for input. The procedure clears the interface and all variables and reads in the parameter values from a file. It either uses the default habitat input file ("_SYSTEM_Habitats.csv") or one defined by the user and *habitats* are created (*CreateHabitatsProc*). It then imports a map, either from an image file (BMP, JPG, GIF or PNG format, no specific dimensions) or from a text file with a specified format (see BEE-SETWARD User Manual). The scaling of the map may be set by the user (or read in from the text file map). Colours are adjusted (*BS_ColourCorrectionsMapProc*) in order to improve the analysis of the map (*BS_AnalyseProc*). Settings are saved (*SaveLoadSettingsProc*) and the new input files are created (*BS_WriteBumbleBeehiveOutProc*). Finally, setup is run again with the new map and input files being loaded.

BS_ImportNewMapProc

```
to BS_ImportNewMapProc [ userInput? ]
  let readFromTextFile? false
  clear-all
  stop-inspecting-dead-agents
  reset-ticks
  if RAND_SEED != 0 [ random-seed RAND_SEED ]

  ReadAllParametersProc
  if user-yes-or-no? "Select a new habitat input file for this map?
                    (If uncertain, press 'No')"
  [ set HabitatsFile FilenameREP true ] ; true: file extension will be reported
  CreateHabitatsProc
  set BorderColor 125.12345678987654 ; a unique color most likely not occurring
  ; on the map (~magenta)

  set OutputWordResult ""
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
ask patches [ set pcolor BorderColor ] ; pcolor set to a unique color,
; to determine the borders of the map, after the map has been loaded
set PatchColoursList []
foreach sort habitats ; saves active habitat types in PatchColoursList
[
  ask ? [ if habitatSwitchedOn?
    [ set PatchColoursList lput habitatColourID PatchColoursList ] ]
]

ifelse userInput? = true
[
  ifelse user-yes-or-no? "Load scan/image file? (click 'No' to load a text file)"
  [
    set MyMap FilenameREP true ; true: file extension will be reported
    import-pcolors MyMap
  ]
  [
    set readFromTextFile? true
    BS_ReadMapFromTextProc ; import-pcolors takes place in that procedure
  ]
]
[ import-pcolors MyMap ] ; (i.e. if userInput? = false)

if readFromTextFile? = false [ BS_Dialogue_ScalingProc ]
BS_ColourCorrectionsMapProc
BS_AnalyseProc
ask patches [set pcolor grey ]
ask patches with [ member? patchColor PatchColoursList = true] [ set pcolor patchColor ]
SaveLoadSettingsProc "Save!" "" ; create new Parameter file
BS_WriteBumbleBeehaveOutProc userInput?
Setup
end
```

FilenameREP

Reports a filename - either with or without extension - based on the path, selected by the user

```
to-report FilenameREP [ extension? ] ; if extension? = true, then also the
; file extension (e.g. ".png") will be reported

let path user-file
set path reverse path ;; as position reports only the first (and not the last)
; occurrence, string is reversed

let cutHere1 0
if extension? = false [ set cutHere1 1 + position "." path ]
let cutHere2 position "\\\" path ; this is the filename + extension, but reversed
set path substring path cutHere1 cutHere2 ; reversed filename without extension
let myFilename reverse path ; this is the actual filename
report myFilename
end
```

CreateHabitatsProc

Creates the habitats from the specifications of the "csv" habitats input file (*HabitatsFile*).

```
to CreateHabitatsProc
ifelse file-exists? HabitatsFile ; e.g. _SYSTEM_Habitats.csv (defined in Parameter.csv)
[
  set HabitatDataCSV csv:from-file HabitatsFile ; read file & save data
; the csv file contains information about which habitat types are represented
; by which colour and the relative abundance of flower species
let headerList item 0 HabitatDataCSV ; the header of the csv table is saved in headerList:
; ["habitatColourID" "colourRangeMin" "colourRangeMax" "habitatSwitchedOn?"
; "habitatType" "ONLY FLOWER DATA BEYOND THIS COLUMN!" "Alsike_clover" "Bugle" ...]
let firstColumnWithFlowerspecies (position "ONLY FLOWER DATA BEYOND THIS COLUMN!"
headerList) + 1 ; the column of the first plant species (probably Alsike_clover)
foreach but-first HabitatDataCSV ; goes through all 'lines' (except of header)
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
[
  let currentFlowerList []
  let myHabitatType "none"
  let column 0
  let nameOfCurrentList first ?
  create-habitats 1
  [
    set habitatColourID item (position "habitatColourID" headerList) ? ; searches for
                                ; "habitatColourID" in the header to identify the column,
                                ; then takes the value of this column for the current data line
    set colourRangeMin item (position "colourRangeMin" headerList) ?
    set colourRangeMax item (position "colourRangeMax" headerList) ?
    set habitatType item (position "habitatType" headerList) ?
    set habitatSwitchedOn? item (position "habitatSwitchedOn?" headerList) ?

    foreach ? ; goes through all 'columns' of this 'line' to collect data
              ; to populate flowerspecieslist:
    [
      let content ? ; the content of a single 'cell'
      if column >= firstColumnWithFlowerspecies and content > 0
      [
        set content precision content 3 ; content here represents the relative
                                        ; abundance of a flower species in the current habitat type
        let speciesNameString (word "[" "\"" item column headerList "\"" )
        set currentFlowerList lput (word speciesNameString " " content "]" )
                                currentFlowerList
      ]
      set column column + 1
    ]
    set flowerspecieslist currentFlowerList
    if nameOfCurrentList = "FlowerSpeciesList_Legume"
    [
      set FlowerSpeciesList_Legume currentFlowerList
      if habitatType != "undefined" [ set Patchtype_Legume (word "\"" habitatType "\"") ]
    ]

    if nameOfCurrentList = "FlowerSpeciesList_Margin"
    [
      set FlowerSpeciesList_Margin currentFlowerList
      if habitatType != "undefined" [ set Patchtype_Margin (word "\"" habitatType "\"") ]
    ]

    if nameOfCurrentList = "FlowerSpeciesList_Plot"
    [
      set FlowerSpeciesList_Plot currentFlowerList
      if habitatType != "undefined" [ set Patchtype_Plot (word "\"" habitatType "\"") ]
    ]
  ] ; end CREATE HABITATS
] ; end "foreach but-first HabitatDataCSV"
] ; end if(else) file-exists?
[ ; ELSE: if file doesn't exist:
  user-message (word "The specified HabitatsFile file cannot be loaded: " HabitatsFile)
]
end
```

BS_ReadMapFromTextProc

Creates the projects map from the information of a text file, which can, for example, be created via ArcGIS. If the defined map is smaller than the model world, then the map will be placed in the centre.

```
to BS_ReadMapFromTextProc
  let textFileToOpen FilenameREP false ; if false: file extension will not be reported
  set MyMap (word textFileToOpen ".png")
  file-open (word textFileToOpen ".txt")
  set GIS_ncols DataFromTextMapREP
  set GIS_nrows DataFromTextMapREP
  set GIS_xllcorner DataFromTextMapREP
  set GIS_yllcorner DataFromTextMapREP
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

set BS_Scaling_m/NLpatches DataFromTextMapREP
set GIS_NoDataValue DataFromTextMapREP
let leftMargin (max-pxcor - GIS_ncols) / 2
let bottomMargin (max-pycor - GIS_nrows) / 2
foreach sort patches
[
  ask ?
  [
    if (pxcor >= leftMargin and pxcor < GIS_ncols + leftMargin)
      and (pycor >= bottomMargin and pycor < GIS_nrows + bottomMargin)
      and file-at-end? = false
      [
        set pcolor file-read
      ]
  ]
]
file-close
ask patches
[
  set originalColor pcolor ;; the color of the (grid) patch in the original file
  set plabel-color black
  set flowerPatchID -1
  set patchColor pcolor ;; saves the color after rounding
]
end

```

DataFromTextMapREP

This reporter-procedure is called, when the map is defined via a text file instead of an image file to read in the header, which defines the dimensions of the map, its real-world coordinates and *GIS_NoDataValue* to denote grid cells which are shown on the NetLogo world but are not defined by the map.

```

to-report DataFromTextMapREP
  ; used when map is read in from TextMap with the new, GIS compatible format,
  ; result is the value in the data line, but not the name of the variable
  let dataline file-read-line
  let datalineRev reverse dataline
  while [ first datalineRev = " " ] [ set datalineRev but-first datalineRev ] ; to get rid of
  ; potential blanks hidden at the end (now beginning!)
  let firstBlankPosition position " " datalineRev
  let resultRev substring datalineRev 0 firstBlankPosition ; incl. position 0,
  ; excludes position 3
  set datalineRev remove " " resultRev
  let result read-from-string reverse datalineRev
  report result
end

```

BS_Dialogue_ScalingProc

This procedure defines the scaling of the map, by linking the the distance of two locations with their their real distance. The user can choose from four options (no changes, if scaling is already defined, length of the x-axis, length of the y-axis or distance between two points chosen by the user) and has then to provide the real distance.

```

to BS_Dialogue_ScalingProc
  set BS_Scaling_m/NLpatches 1 / Scaling_NLpatches/m ; scaling in BEESCOUT is
  ; 1 / scaling in Bumble-BEEHAVE

  let distance_NLpatches 0
  let p1 nobody
  let p2 nobody
  let memoColorP1 0
  let memoColorP2 0
  let userChoice1 user-one-of "Choose a method to define the scaling of your map"
  [
    "Scaling is already defined - no changes needed"
  ]
end

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
"Direct Input"
"Distance left - right side"
"Distance top - bottom"
"Distance any two points"
]

if userChoice1 = "Distance left - right side"
    [ set distance_NLpatches max-pxcor - min-pxcor ]
if userChoice1 = "Distance top - bottom" [ set distance_NLpatches max-pycor - min-pycor ]
if userChoice1 = "Distance any two points" ; choose 2 points via mouse clicks:
[
    let userHappy? false
    let pointColor red
    user-message "Please select now two reference points of known distance via two mouse
        clicks (make sure speed-slider is set to 'normal speed')"
    while [ userHappy? = false ]
    [
        while [ p1 = nobody ]
        [
            if mouse-down?
            [
                set p1 patch mouse-xcor mouse-ycor
                ask p1
                [
                    set memoColorP1 pcolor
                    set pcolor pointColor
                ]
            ]
            display
        ]
        wait 0.5
        while [ p2 = nobody ]
        [
            if mouse-down?
            [
                set p2 patch mouse-xcor mouse-ycor
                ask p2 ; patch mouse-xcor mouse-ycor
                [
                    set memoColorP2 pcolor
                    set pcolor pointColor
                ]
            ]
            display
        ]
        let userChoiceHappiness user-one-of "Are you happy with your reference points?"
        [
            "Yes - continue"
            "No - do it again"
            "No - do it again with different point colors"
        ]

        if userChoiceHappiness = "Yes - continue" [ set userHappy? true ]
        if userChoiceHappiness = "No - do it again"
            or userChoiceHappiness = "No - do it again with different point colors"
        [
            ask p1 [ set pcolor memoColorP1 ]
            ask p2 [ set pcolor memoColorP2 ]
            set p1 nobody
            set p2 nobody
            set memoColorP1 0
            set memoColorP2 0
            display
        ]
        if userChoiceHappiness = "No - do it again with different point colors"
        [
            set pointColor read-from-string user-one-of "Choose the colour of your
                reference points:"
            [
                "Black"
                "White"
                "Grey"
                "Blue"
                "Green"
                "Yellow"
                "Red"
            ]
        ]
    ]
]
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

        "Green"
    ]
]
] ; while user unhappy
ask p1 [ set distance_NLpatches distance p2 ]
] ; "Distance any two points"

if userChoice1 != "Scaling is already defined - no changes needed"
    and userChoice1 != "Direct Input"
    [ set BS_ScaleDistance_m read-from-string user-input "Real distance [m] is (e.g. 150): " ]
if userChoice1 = "Distance any two points"
[
    ask p1 [ set pcolor memoColorP1 ]
    ask p2 [ set pcolor memoColorP2 ]
]
if userChoice1 != "Scaling is already defined - no changes needed"
    and userChoice1 != "Direct Input"
    [ set BS_Scaling_m/NLpatches BS_ScaleDistance_m / distance_NLpatches ] ;; real distance [m]
                                                ; divided by distance of grid points
if userChoice1 = "Direct Input"
    [ set BS_Scaling_m/NLpatches read-from-string user-input "Scaling is [m/grid cell]
                                                                (e.g. 25): " ]
end

```

BS_ColourCorrectionsMapProc

Each habitat type is represented on the map by a specific colour (*habitatColourID*). When a map is read in from an image file, for each grid cell it has to be decided which (if any) habitat type it represents, which requires a range of colour shades being interpreted as a certain habitat type. This range of colours is defined in the habitat input file by *colourRangeMin* and *colourRangeMax*. The *BS_ColourCorrectionsMapProc* sets the colour of all gridcells that are within the colour range of a certain habitat type to *habitatColourID* of this habitat type. For example: all yellowish colours in the original image are set to yellow, which might represent oilseed rape.

to BS_ColourCorrectionsMapProc

```

ask patches
[
    set originalColor pcolor ;; the color of the (grid) patch in the original file
    set plabel-color black
    set flowerPatchID -1
    if remainder pcolor 10 < Black_th [ set pcolor black ]
    if remainder pcolor 10 > White_th [ set pcolor white ]
]

ask habitats
[
    let memoColorID habitatColourID
    let memoColorMin colourRangeMin
    let memoColorMax colourRangeMax

    ask patches with [ pcolor >= memoColorMin and pcolor <= memoColorMax
                      and (pcolor != BorderColor) ]
        [ set pcolor memoColorID ]
]
ask patches [ set patchColor pcolor ]
end

```

BS_AnalyseProc

This procedure is a modified version of BEESCOUT's *AnalyseProc*. It identifies food patches (connected areas of the same colour), their size, location and habitat type. For more details, see ODD protocol of BEESCOUT.

to BS_AnalyseProc

```

ask turtles with [ breed != habitats ] [ die ]
ask turtles [ hide-turtle ]

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

let currentColor 0
let currentPatchID -1
let flowerPatchCounter 0
set Repetitions round (MaxPatchRadius_m / BS_Scaling_m/NLpatches) ; # of repetition depends
                                                                    ; on the scale of the landscape

foreach sort patches
[
  ask ?    ;; determines flowerpatches: same-coloured (nlogo-)patches with ID -1
           ; are searched, it gets a new ID and all connected (nlogo-)patches
           ; with the same colour get the same ID
  [
    if member? patchColor PatchColoursList
    [
      if flowerPatchID < 0 ;; if patch is not identified yet (flowerPatchID is
                          ; patches-own variable, set to -1 in BS_ColourCorrectionsMapProc)
      [
        set flowerPatchID flowerPatchCounter
        set currentColor patchcolor ;; colour of the flower patch is the colour
                                    ; of the "firstPatchOfFlowerpatch"
        set firstPatchOfFlowerpatch true
        set currentPatchID flowerPatchID

        let search-patches patches in-radius Repetitions
        repeat Repetitions
        [
          ask search-patches with [(patchcolor = currentColor)
                                  and (flowerPatchID = currentPatchID)] ;; connected (nlogo-)patches around
                                  ; the firstPatchOfFlowerpatch are searched and defined as part of this
                                  ; flower patch
          [
            ask neighbors with [(patchcolor = currentColor) and (flowerPatchID = -1)]
            [ set flowerPatchID currentPatchID ]
          ]
        ]
        set flowerPatchCounter flowerPatchCounter + 1
      ]
    ]
    set pcolor lime ; to show progress of map analysis
  ]
  display
]

ask patches with [ flowerPatchID >= 0 ] [ set flowerPatchID flowerPatchID + count turtles ]
; in the original BEESCOUT model, all turtles were cleared at the beginning of
; this procedure, now, habitat-turtles are not cleared, hence flowerPatchID needs
; to be increased by the number of (habitat) turtles

set Npatches currentPatchID + 1 ; as 1st patch has id 0
BS_CreatePatchStatisticsProc    ;; creates "patchStatistics" (turtles) to store data
                                ; of the flower patches
BS_DetermineSizeProc
end

```

BS_CreatePatchStatisticsProc

Creates *patchStatistics* (see also BEESCOUT ODD), agents that store the location size and habitat type of each food patch that has been identified in *BS_AnalyseProc*. The location of the patch is calculated from the average x- and y-coordinates of all grid cells, which are part of this food patch (see *BS_AnalyseProc*). If the calculated location lies outside the actual area of the food patch, then it is moved to the closest location within the food patch.

```

to BS_CreatePatchStatisticsProc
  let currentXcor 0
  let currentYcor 0
  let currentWho 0
  let currentColor 0

```


Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

create-PatchStatistics Npatches ; Npatches is set in BS_AnalyseProc and equals
                                ; the number of identified food patches

[
  set size 2
  set shape "circle"
  set currentWho who
  set patchInfo "no info"
  ; define the patch location by calculating the mean x and y-coordinates
  ; of the gridcells (Netlogo patches) that are part of this food patch:
  set xcor mean [ pxcor ] of patches with [ flowerPatchID = currentWho ]
  set ycor mean [ pycor ] of patches with [ flowerPatchID = currentWho ]
  set currentColor patchcolor
  if currentWho != flowerPatchID ; i.e. if calculated location is outside of the
                                ; actual food patch area:

    [
      ask min-one-of patches with [ flowerPatchID = currentWho ] [distance myself]
      [
        set currentXcor pxcor
        set currentYcor pycor
        set currentColor patchcolor
      ]
      setxy currentXcor currentYcor
    ]
  set color currentColor - 1
  set label-color white
  set label who
  let memoFlowerSpeciesList []
  let memoPatchType -1
  let patchcolorMemo patchcolor
  ask habitats with [ habitatColourID = patchcolorMemo and habitatSwitchedOn? = true ]
  [
    set memoFlowerSpeciesList flowerspecieslist
    set memoPatchType habitatType ; NOTE: this flowerspecieslist is
                                ; the habitats-own variable
  ]
  set patchType memoPatchType
  set flowerSpeciesList memoFlowerSpeciesList ; NOTE: here, flowerspecieslist
                                              ; is a patchstatistics-own variable
]
end

```

BS_DetermineSizeProc

Determines the area and the perimeter of each food patch identified in *BS_AnalyseProc*.

```

to BS_DetermineSizeProc
  let currentWho 0
  foreach sort patchStatistics
  [
    ask ?
    [
      set currentWho Who
      set areaPx (count patches with [flowerPatchID = currentWho])
      set areaSqm round(areaPx * BS_Scaling_m/NLpatches * BS_Scaling_m/NLpatches)
      set perimeter_m BS_PerimeterREP
    ]
  ]
end

```

BS_PerimeterREP

Reports the approximate perimeter of a food patch. For each grid cells of this food patch, the number of neighbouring grid cells that belong to no or another food patch is summed up in the local variable *borderCells*. The perimeter is then assumed to equal the number of *borderCells* multiplied by their edge length (i.e. by *BS_Scaling_m/NLpatches*)

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

to-report BS_PerimeterREP

```
let whoPatch who
let borderCells 0
ask patches with [ flowerPatchID = whoPatch ]
[
  let pcolorMemo patchcolor
  if count neighbors with [ patchcolor != pcolorMemo ] > 0
    [ set borderCells borderCells + 1 ]
]
set perimeter_m borderCells * BS_Scaling_m/NLpatches
report perimeter_m
end
```

BS_WriteBumbleBeehaveOutProc

Writes the scaling, the total number of identified food patches and all the data for each food patch into a text file (*FoodsourcesFile*, i.e. "name_foodsources.txt"). Also creates a png image file (*MyMap*) from the NetLogo "world". If *userInput?* is true, the user will be asked to provide a project name. For changes in the format of *FoodsourcesFile* see chapter "INPUT FILES" at the end of this document.

to BS_WriteBumbleBeehaveOutProc [userInput?]

```
let userName ""
ifelse userInput? = true
[
  let userNamePrelim user-input (word "Name of your map: " remove "_SYSTEM_"
    substring MyMap 0 (length MyMap - 4) ".png" " (leave blank to not change name or
    insert new name (no extension, no quotation marks)): ")
  if length userNamePrelim > 4 and item (length userNamePrelim - 4) userNamePrelim = "."
    ; item (length userNamePrelim - 4) defines the position of the dot,
    ; as file extension is 3 characters long
    [ set userNamePrelim substring userNamePrelim 0 (length userNamePrelim - 4)]
    ; removes file extension

  ifelse userNamePrelim = ""
  [ set userName substring MyMap 0 (length MyMap - 4)]
  [ set userName userNamePrelim ]
]
[ set userName remove ".png" MyMap ]
set userName (word remove "_SYSTEM_" userName) ; to make sure that system files
; are not overwritten!

set FoodsourcesFile (word userName "_Foodsources.txt")
set MyMap (word userName ".png") ; the name of the png image file that is going to be created
ask turtles [ hide-turtle ] ; we don't want to see bees etc. on the map!
export-view (word userName ".png") ; here, the actual map image is created
if file-exists? FoodsourcesFile [ file-delete FoodsourcesFile ]
; a new file will be created - so first delete the old one!

file-open FoodsourcesFile
file-print BS_Scaling_m/NLpatches ; 1. line: only one value, the scaling
file-print count patchStatistics ; 2. line: only one value, tnumber of food patches
;; NOTE: file format was modified for Beestew, NEW: perimeter_m
;; REMOVED: quantityPollen_g proteinPollenProp quantityNectar_l concentration_mol/l startDay
;; stopDay corollaDepth_mm nectarFlowerVolume_myl intFlowerTime_s

file-print "id patchType patchColour xcor ycor size_sqm flowerSpeciesList perimeter_m info"
; 3. line: this is the header

foreach sort patchStatistics
[
  ask ? ; now the actual data for each food patch are written in the file:
  [
    file-type (who - count habitats) file-type " " ; number of habitat-turtles subtracted,
    ; to make sure, ID listed in foodsource text file is identical to "who" of
    ; that foodsource
    file-type "\""
    file-type patchType
    file-type "\""
    file-type " "
    file-type color + 1 file-type " "
    file-type precision xcor 3 file-type " "
  ]
]
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

        file-type precision ycor 3 file-type " "
        file-type precision areaSqm 1 file-type " "
        file-type flowerSpeciesList file-type " "
        file-type perimeter_m file-type " "
        file-write patchInfo ; printed as string i.e. with " "
        file-print ( " " )
    ]
]
file-close
end

```

SaveLoadSettingsProc

This procedure either updates or creates a new parameter input file ("*name_Parameters.csv*"), using the current setting or loads an existing parameter input file to update the setting. User input may be required.

```

to SaveLoadSettingsProc [ SaveOrLoad? suffix ] ; SaveOrLoad?: "SaveOrLoad!" "Save!" "Load!"
    ; suffix: either "" or set by user to specify map swith CSO, e.g. "margins"
    ; (see StewardshipOptionsUpdateFoodSourcesProc)
let choice ""
if SaveOrLoad? = "SaveOrLoad!"
    [ set choice user-one-of "Do you want to save the current setting for this particular map?
        Or would you like to import the setting from another map?" ["Save current setting"
            "Import from another map/Parameters file"]]
if SaveOrLoad? = "Load!" [ set choice "Import from another map/Parameters file" ]
if SaveOrLoad? = "Save!" [ set choice "Save current setting" ]
if choice = "Save current setting"
    [
        let header ""
        ifelse file-exists? "_SYSTEM_Parameters.csv"
        [
            file-open "_SYSTEM_Parameters.csv"
            set header csv:from-row file-read-line
            file-close
        ]
        [ user-message "Can't find input file '_SYSTEM_Parameters.csv'" ]

let parameterValues [] ; this will create line 2 in the parameter file
foreach header ; header is a list of all parameters that are defined in the parameter file
    ; i.e. AbundanceBoost BeeSpeciesInitialQueensListAsString Backgroundcolour...
    [
        let nextCommand (word "set MyValue " ? ) ; problem here ? refers to a string
            ; but may actually represent either a string or a number
run nextCommand ; runs a command, e.g. set MaxHibernatingQueens 10000
ifelse is-string? MyValue = true ; if string, "" needs to be added!:
    ; (MyValue: global variable, as "run" cannot access local variable)
    [ set parameterValues lput (word "\" MyValue "\"") parameterValues ]
    [ set parameterValues lput MyValue parameterValues ]
    ]
let newParametersList []
set newParametersList lput header newParametersList
set newParametersList lput parameterValues newParametersList

if behaviorspace-run-number = 0 ; if run via BehaviorSpace then behaviorspace-run-number
    ; is > 0 and MyParametersFile is not renamed here! This is important, as a specific
    ; parameter files needs to be created for each BehavSpace run (see SetupBehaviorSpace)
[
    set MyParametersFile (word remove "_SYSTEM_" MyMap) ; make sure, system files cannot
        ; be overwritten!
if item (length MyParametersFile - 4) MyParametersFile = "."
    [ set MyParametersFile substring MyParametersFile 0 (length MyParametersFile - 4)]
        ; removes file extensions
set MyParametersFile (word remove suffix MyParametersFile)
set MyParametersFile (word MyParametersFile suffix "_Parameters.csv")
]
csv:to-file MyParametersFile newParametersList
]

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

; to LOAD a parameter file:
if choice = "Import from another map/Parameters file" ; imports the parameter file
                                                    ; (but not the map) from another project

[
  let myChosenParameterfile user-file ; reports the complete path - however,
                                      ; only the filename is required!
  ifelse member? "_Parameters.csv" myChosenParameterfile ; chosen file is a parameter file
  [
    set myChosenParameterfile reverse myChosenParameterfile ;; as position reports only
                                                              ; the first (and not the last) occurrence, string is reversed
    let cutHere position "\\\" myChosenParameterfile ; this is the filename + extension,
                                                         ; but reversed
    set myChosenParameterfile substring myChosenParameterfile 0 cutHere ; reversed
                                                              ; filename without extension
    set myChosenParameterfile reverse myChosenParameterfile ; this is the actual filename
  ]
  [
    ; the chosen file is not a parameter file but a map or foodsources file
    set myChosenParameterfile reverse myChosenParameterfile ;; as position reports only
                                                              ; the first (and not the last) occurrence, string is reversed
    let cutHere position "\\\" myChosenParameterfile ; this is the filename + extension,
                                                         ; but reversed
    set myChosenParameterfile substring myChosenParameterfile 4 cutHere ; reversed
                                                              ; filename without extension
    set myChosenParameterfile reverse myChosenParameterfile ; this is the actual filename
    set myChosenParameterfile remove "_Foodsources" myChosenParameterfile ; this is the
                                                                              ; correct map name
    set myChosenParameterfile (word myChosenParameterfile suffix "_Parameters.csv")
  ]
  ; myChosenParameterfile has now the correct parameter file name
  ifelse file-exists? myChosenParameterfile
  [
    let memoOwnParameterfile? ProjectsOwnParameterFile?
    set ProjectsOwnParameterFile? false ; needs to be false to load a different
                                         ; parameters file

    set MyParametersFile myChosenParameterfile
    user-message (word "Load this parameter file: " MyParametersFile)
    ReadAllParametersProc ; the new parameter settings are loaded
    Setup
    set ProjectsOwnParameterFile? memoOwnParameterfile?
  ]
  [ user-message "No Parameter file is linked to the chosen map!" ]
]
end

```

Button 3 "Set Parameter Values"

Calls the procedure *ParametersSetManuallyProc*.

ParametersSetManuallyProc

Opens a dialogue from which the user can choose the parameter they want to change and then provide the new parameter value. Finally the new setting is saved.

```

to ParametersSetManuallyProc ; called by button
  foreach AllParametersList
  [
    let parameter remove " " ? ; some parameters have a blank added to the end of their name,
                                ; which is removed here
    let command (word "set GenericRunCommandValue " parameter)
    run command
    output-print (word parameter ": " GenericRunCommandValue)
  ]

  let parameterType "undefined"
  let parameterToBeChanged user-one-of "Select parameter you would like to change (press

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

        'Setup' when finished): "      ; NOTE: only a selection of parameters are shown here
[
  "AbundanceBoost"
  "BeeSpeciesInitialQueensListAsString" ; e.g. "Bee_longTongue 500 Bee_shortTongue 500"
  "COLONIES IBM"
  "CumulVisitsOnlyLastYear?"
  "FlowerspeciesFile"
  "FoodSourceLimit"
  "ForagingMortalityFactor"
  "ForagingMortalityModel"
  "Gridsize"
  "HabitatsFile"
  "KeepDeadColonies?"
  "Lambda_detectProb"
  "MapAreaIncluded"
  "MasterSizeFactor"
  "MaxForagingRange_m"
  "MaxHibernatingQueens"
  "MaxPatchRadius_m"
  "MergeHedges?"
  "MinSizeFoodSources?"
  "N_Badgers"
  "N_Psithyrus"
  "RemoveEmptyFoodSources?"
  "SexLocus?"
  "ShowCohorts?"
  "ShowDeadCols?"
  "ShowFoodsources?"
  "ShowGrid?"
  "ShowInspectedColony?"
  "ShowMasterpatchesOnly?"
  "ShowNests?"
  "ShowPlots?"
  "ShowQueens?"
  "ShowSearchingQueens?"
  "ShowWeather?"
  "BeespeciesFile"
  "StopExtinct?"
  "UnlimitedMales?"
  "Weather"
]
let newValue user-input "Set new value for this parameter (leave blank to cancel)"
if newValue != "" [ run (word "set " parameterToBeChanged " " newValue) ] ; here the
                                ; parameter value is changed
SaveLoadSettingsProc "Save!" "" ; new settings are saved
end

```

Button 4 "Load Setting"

Calls the procedure *SaveLoadSettingsProc* twice, first with "Load!" as input, then again with "Save!" as input. For a description of *SaveLoadSettingsProc*, see above ("Button 2", panel "Maps and Settings").

Button 5 " Save Setting"

Calls the procedure *SaveLoadSettingsProc*, with "Save!" as input. For a description of *SaveLoadSettingsProc*, see above ("Button 2", panel "Maps and Settings").

Button 6 "Show or hide Scale Bar"

Calls the procedure *ButtonScaleBarProc* to show or remove a black&white scale bar on the map.

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

ButtonScaleBarProc

Creates two NetLogo agents of the breed *sign*. The first one is a black and white line, representing the scale bar (shape = "scalebar"). The second one is invisible (shape = "scaledistance") and only used to show a label with the real distance represented by the scale bar. Lengths of the scale bar and represented distances are chosen in a way to avoid too long or short scale bars or too odd numbers for the distance. If the scale bar and label already exist, they are deleted ("die").

```
to ButtonScaleBarProc
  let xpos 1      ; xy location of the left end of the scale bar ("scalebar" is only a...
  let ypos 207    ; "half line" hence with increasing size it only extends on the right side)
  let targetSize max-pxcor / 2.5 ; approximate length of the bar
                        ; size 120 i.e. actual length: 60
  let targetLength_m targetSize / SCALING_NLpatches/m / 2
  let divisor (10 ^ (round (log targetLength_m 10))) / 2 ; to avoid too odd numbers
  let realBarLength_m divisor * ceiling (targetLength_m / divisor)
  ifelse count signs with [ shape = "scalebar" or shape = "scaledistance"] > 0
    [ ask signs with [ shape = "scalebar" or shape = "scaledistance" ] [ die ] ]
    [
      create-Signs 2
      [
        set color black
        set shape "scalebar"
        set size 2 * SCALING_NLpatches/m * realBarLength_m
        setxy xpos ypos
      ]
      ask one-of signs with [ shape = "scalebar" ] ; shows the label, e.g. "1000 m"
      [
        set shape "scaledistance"
        set size 1
        setxy xpos + 13 ypos - 4
        set label-color black
        set label (word realBarLength_m " m")
        if realBarLength_m >= 10000 ; switch units from m to km if needed
          [ set label (word (realBarLength_m / 1000) " km") ]
      ]
    ]
end
```

Button 7 "Initial Queens"

Calls the procedure *ButtonInitialQueensProc* to set the initial number of queens for *B. terrestris* and/or all bumblebee species that have been defined.

ButtonInitialQueensProc

The user is asked if only *B. terrestris* should be present and, if this is the case, their initial number of queens can be set. If also other bumblebee species should be present, user-dialogues address all species listed in *BeeSpeciesDefinedList* and for each one, the user is asked if this particular species should be present and if so, with how many initial queens. The information about which species are present and their initial queen numbers are saved in the local variable *queensCommandString*. Then the global variable *BeeSpeciesInitialQueensListAsString* is set to *queensCommandString* and the procedure *SaveLoadSettingsProc* is called to save the current setting in the parameter file of the project and finally, *Setup* is called.

```
to ButtonInitialQueensProc ; Button "Initial Queens"
  let queensCommandString ""
  let terrestrisOnly? user-yes-or-no? "Set queens for B. terrestris only?" ; user input
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
ifelse terrestrisOnly? = true ; if ONLY B. terrestris is present:
[
  let nTerrestris user-input "Initial number of B. terrestris queens: "
  set BeeSpeciesInitialQueensListAsString (word "B_terrestris " nTerrestris)
]
[
  foreach BeeSpeciesDefinedList ; if more or other species than B. terrestris are added:
  [
    let addThisSpecies? user-yes-or-no? (word "Are there any " ? " queens present?")
    if addThisSpecies? = true
    [
      let queensThisSpecies read-from-string user-input
                                (word "How many " ? " queens are present?")
      set queensCommandString (word queensCommandString ? " " queensThisSpecies " ")
    ]
  ]
  set BeeSpeciesInitialQueensListAsString queensCommandString
]
SaveLoadSettingsProc "Save!" ""
Setup
end
```

Panel "My Own Maps"

This panel allows the user to easily load previously created projects.

Buttons 1-6 "Load Map 1(..6)"

These buttons set *MyMap* to *MySavedMap1(..6)*. Then the procedures *ClearProc* and *SetupWithoutClearingProc* are called. If *MySavedMap1(..6)* is "", then *MySavedMap1(..6)* is set to *MyMap* i.e. the current project is linked to this button and will be called next time the button is clicked.

Button 7 "Delete one of My Maps"

Opens a user dialogue to determine which button of this panel (or all of them) should be cleared and *MySavedMap* for this button is set to "". Then *PanelSettingProc* is called (see above, "Setup Procedures").

Panel "Modify Maps"

This panel allows the user to easily modify maps by drawing on them or replacing habitat types of one or several food patches.

Button 1 "Choose colour"

Calls the procedure *ButtonChooseColourProc*.

ButtonChooseColourProc

ButtonChooseColourProc allows the user to change the colour (*SetColour*) of the brush. If "Restore" is chosen as colour, the brush acts as eraser on recently added drawing. At the end, "DrawProc" is called (see above).

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

to ButtonChooseColourProc

```

let textUser "Choose a colour: " ; informs the user which habitat type (if any) is
; represented by each colour (only if habitat is already represented on map)

carefully
  [ set textUser (word textUser
    " Pink: " [ patchtype ] of one-of foodsources with [ color = Pink - 1 ])]
  [ set textUser (word textUser " Pink: undefined")]
carefully
  [ set textUser (word textUser
    "; Magenta: " [ patchtype ] of one-of foodsources with [ color = Magenta - 1 ])]
  [ set textUser (word textUser "; Magenta: undefined")]
carefully
  [ set textUser (word textUser
    "; Brown: " [ patchtype ] of one-of foodsources with [ color = Brown - 1 ])]
  [ set textUser (word textUser "; Brown: undefined")]
carefully
  [ set textUser (word textUser
    "; Blue: " [ patchtype ] of one-of foodsources with [ color = Blue - 1 ])]
  [ set textUser (word textUser "; Blue: undefined")]
carefully
  [ set textUser (word textUser
    "; Green: " [ patchtype ] of one-of foodsources with [ color = Green - 1 ])]
  [ set textUser (word textUser "; Green: undefined")]
carefully
  [ set textUser (word textUser
    "; Yellow: " [ patchtype ] of one-of foodsources with [ color = Yellow - 1 ])]
  [ set textUser (word textUser "; Yellow: undefined")]
carefully
  [ set textUser (word textUser
    "; Red: " [ patchtype ] of one-of foodsources with [ color = Red - 1 ])]
  [ set textUser (word textUser "; Red: undefined")]
carefully
  [ set textUser (word textUser
    "; Turquoise: " [ patchtype ] of one-of foodsources with [ color = Turquoise - 1 ])]
  [ set textUser (word textUser "; Turquoise: undefined")]
carefully
  [ set textUser (word textUser
    "; Violet: " [ patchtype ] of one-of foodsources with [ color = Violet - 1 ])]
  [ set textUser (word textUser "; Violet: undefined")]
set SetColour user-one-of textUser ; colour options for the user
[
  "Pink"
  "Magenta"
  "Brown"
  "Blue"
  "Green"
  "Yellow"
  "Red"
  "Turquoise"
  "Violet"
  "Orange"
  "Lime"
  "Cyan"
  "Sky"
  "Grey"
  "Black"
  "White"
  "Restore"
  "Define by number"
]
if SetColour = "Define by number" [ set SetColour user-input
  "Please define a colour, using the NetLogo colour scheme [0..140[" ]
set Button1Monitor (word "Choose Colour (" SetColour ")")
DrawProc
end

```

Button 2 "Set Brush Size"

Calls the procedure *ButtonBrushSizeProc*.

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

ButtonBrushSizeProc

ButtonChooseColourProc allows the user to set the brush size. Afterwards, the procedure *DrawProc* is called (see above).

```
to ButtonBrushSizeProc
  set BrushSize read-from-string user-input "Set brush size (value between 1 - 200)"
  if BrushSize < 1 [ set BrushSize 1 ]
  if BrushSize > 100 [ set BrushSize 200 ]
  ifelse CircularBrush? = true ; recalculate area covered by brush:
    [ set BrushArea_ha (pi * ((BrushSize / Scaling_NLpatches/m) / 2) ^ 2) / 10000 ] ; circular
    [ set BrushArea_ha (BrushSize / Scaling_NLpatches/m) ^ 2 / 10000 ] ; square
  set Button4Monitor (word "Set Brush Size (" BrushSize ")
                        (ca. " precision BrushArea_ha 1 "ha)")
  DrawProc
end
```

Button 3 "Switch Brush Shape to ..."

Allows the user to switch between a circular and a square brush shape. Calls the procedure *DrawProc* afterwards (see above).

(*PanelButtonProc*):

```
if myButtonCommand = "Switch Brush Shape to Circular"
  [
    set CircularBrush? true
    set Button5Monitor "Switch Brush Shape to Square"
    DrawProc
  ]
if myButtonCommand = "Switch Brush Shape to Square"
  [
    set CircularBrush? false
    set Button5Monitor "Switch Brush Shape to Circular"
    DrawProc
  ]
```

Button 4 "Draw on Map"

Calls the procedure *DrawProc*.

DrawProc

Allows the user to draw on the map. The colour, size and shape of the brush can be changed with the buttons 3 - 5.

```
to DrawProc
  let finished? false ; set to true, when the mouse pointer leaves the map
  let startTime timer
  while [ finished? = false ] ; this loop is continuously run while the mouse pointer remains
                                ; on the map
  [
    let clickedXcor mouse-xcor ; current position of the mouse
    let clickedYcor mouse-ycor
    ask turtles with [ breed != brushSigns ] [ hide-turtle ] ; hide all bees etc.
    ask brushSigns
    [
      setxy mouse-xcor mouse-ycor ; brush is located at the location of the mouse
      set size BrushSize ; brush sign gets the right size
    ]
  ]
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

    ifelse CircularBrush? = true ; brush shaoe either circular or square
    [ set shape "circleline2" ] ; "circle"
    [ set shape "squareline" ] ; "square"
    show-turtle ; brush becomes visible
    if SetColour != "Restore"
    [ set color read-from-string SetColour ]
    display
  ]
if mouse-down? ; the user starts drawing!
[
  let currentColor grey
  if SetColour != "Restore" ; SetColour is chosen by user (button 3)
  ; "Restore" erases unsaved drawings
  [ set currentColor read-from-string SetColour ]
  ifelse CircularBrush? = true
  [ ; circular brush
    ask patches with [ distancexy (round clickedXcor) (round clickedYcor)
      <= (BrushSize / 2) ]
    [
      ifelse SetColour = "Restore"
      [ set pcolor pcolorSave ] ; erasing
      [ set pcolor currentColor ] ; drawing
    ]
  ]
  [ ; square brush
    ask patches with [ (pxcor >= clickedXcor - (BrushSize / 2)
      and pxcor <= clickedXcor + (BrushSize / 2))
      and (pycor >= clickedYcor - (BrushSize / 2)
      and pycor <= clickedYcor + (BrushSize / 2)) ]
    [
      ifelse SetColour = "Restore"
      [ set pcolor pcolorSave ] ; erasing
      [ set pcolor currentColor ] ; drawing
    ]
  ]
]
ifelse CircularBrush? = true ; calculate area covered by brush size
[ set BrushArea_ha (pi * ((BrushSize / Scaling_NLpatches/m) / 2) ^ 2) / 10000 ] ; circle
[ set BrushArea_ha (BrushSize / Scaling_NLpatches/m) ^ 2 / 10000 ] ; square brush
ask brushSigns [ if mouse-inside? = false [ hide-turtle ] ] ; make sure brush is hidden,
; once "Modifying" is finished
if mouse-inside? = false and timer - startTime > 2 ; user has 2s to move the mouse inside
; the map at the beginning. Moving it out again end "Drawing"
[ if mouse-inside? = false [ set finished? true ] ]
] ; end of "while"
end

```

Button 5 "Replace Colours"

Calls the procedure *ButtonReplaceColoursProc*.

ButtonReplaceColoursProc

Allows the user to change the colour of a food patch. The procedure shows two colour scales from which the user can pick the new colour. They then select the food patch they want to change and can then decide whether only the selected food patch will be modified or all food patches of that habitat type.

```

to ButtonReplaceColoursProc
  let color1 -999
  let color2 -999
  ask turtles [ hide-turtle ] ; hide bees, colonies etc.
  ask patches [ set pcolorsave pcolor ] ; save the current colour of each grid cell
  ; creates 2 colour scales:
  ask patches with [ pxcor >= (max-pxcor - 10)]
  [ set pcolor 140 * (pycor / max-pycor)]
  ask patches with [ pxcor <= 10]
  [

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

    let newColor 140 * (pycor / max-pycor)
    set newColor round (newColor / 10)
    set pcolor (newColor * 10) + 5
  ]
  user-message "Select the NEW colour by clicking on a patch or the colour scale"
  while [ color2 = -999 ]
  [
    if mouse-down?      ; new colour is selected from map/colour scale by clicking on it
    [ set color2 [ pcolor ] of patch mouse-xcor mouse-ycor ]
    display
  ]
  ask patches [ set pcolor pcolorsave ] ; colour scales are no longer shown

  let replaceAllPatches? user-yes-or-no? "Click 'Yes' to replace all patches
    or 'No' to replace only chosen patch, then select a patch by clicking on it."
  let finished? false
  let patchesToBeChecked [] ; grid cells which are neighbours of a grid cell
    ; that has its colour changed
  let patchesToBeCheckedUpdate []
  let chosenNLpatch nobody
  while [ finished? = false ]
  [
    if mouse-down? and chosenNLpatch = nobody
    [
      set chosenNLpatch patch mouse-xcor mouse-ycor
      ask chosenNLpatch
      [
        set color1 pcolor ; saves the colour of the chosen grid cell
        set pcolor color2 ; the colour of this grid cell is changed to the new colour here
        ask neighbors4 [ set patchesToBeChecked lput self patchesToBeChecked ]
          ; self reports the asking agent (in contrast to myself,
          ; which would report chosenNLpatch)
      ]
    ] ; end mouse-down

    ifelse replaceAllPatches? = true ; if true, all grid cells of the same colour
      ; as the chosen one change their colour
    [
      ask patches with [ pcolor = color1 ][ set pcolor color2 ]
      if color1 >= 0 [ set finished? true ] ; initial value of color1: -999
    ]
    [
      foreach patchesToBeChecked ; (true) neighbors are checked (repeatedly)
        ; whether they have the same color
      [
        ask ?
        [
          if pcolor = color1
          [
            set pcolor color2 ; neighbours of grid cells of the same colour change their colour
            ask neighbors4 [ set patchesToBeCheckedUpdate lput self patchesToBeCheckedUpdate ]
            ; self reports the asking agent (in contrast to myself, which would report chosenNLpatch)
          ]
        ]
      ]
      set patchesToBeCheckedUpdate patchesToBeCheckedUpdate
      set patchesToBeCheckedUpdate []
      if chosenNLpatch != nobody and patchesToBeChecked = [] ; no more new neighbours of
        ; the same colour - stop the process!
      [ set finished? true ]
    ]
  ]
  if color1 = color2
  [
    set finished? true ; no need to do anything in this case!
    user-message "Identical colours - no changes were made!"
  ]
] ; end WHILE finished? = false
ask patches [ set plabel "" ]
end

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

Button 6 "Clear whole Map"

This button creates an empty (grey) map (*PanelButtonProc*):

```
;; PANEL: "MODIFY MAPS"
if myButtonCommand = "Clear whole Map"
[
  ask turtles [ hide-turtle ]
  ask patches [ set pcolor grey ]
]
```

Button 7 "Update Current Map"

Calls the procedure *ButtonUseCurrentMapProc*.

ButtonUseCurrentMapProc

This procedure updates the settings and saves the changes made to the map. It calls the procedures *SaveLoadSettingsProc* (see panel "Maps and Settings", button 5) and *BS_ImportNewMapProc* (see panel "Maps and Settings", button 2).

```
to ButtonUseCurrentMapProc
  let myName (word remove ".png" MyMap)
  let prelimName user-input "Set a new name for your map (or leave blank to overwrite): "
  if prelimName != ""
    [ set myName prelimName ] ; if no new name is provided, file names remain the same
  set myName (word remove "_SYSTEM_" myName) ; make sure, system files cannot be overwritten!
  set MyMap (word myName ".png")
  ask turtles [ hide-turtle ] ; we don't want to see bee etc. on the map!
  export-view (word myName ".png") ; image file of the map is created
  SaveLoadSettingsProc "Save!" "" ; settings are saved
  BS_ImportNewMapProc FALSE ; FALSE: no user input required;
end
```

Panel "Stewardship Options"

This panel allows the user to add stewardship options to selected fields.

Button 1 "Select Stewardship Option"

Ask the user to choose, which stewardship option will be applied.

(*PanelButtonProc*):

```
if member? "Select Stewardship Option" myButtonCommand
[
  set StewardshipOption user-one-of "Select a Stewardship Option" [ "legume" "margin" "plot" ]
  set Button3Monitor (word "Select Stewardship Option (" StewardshipOption ")")
]
```

Button 2 "Select Field"

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

Calls the procedure *ButtonSelectFieldProc*.

ButtonSelectFieldProc

Allows the user to select crop fields on the map for stewardship options to be applied.

```
to ButtonSelectFieldProc
  let finished? false
  ask signs with [ shape = "circletarget" ] [ show-turtle ] ; pointer becomes visible
  let startTime timer
  while [ finished? = false ]
  [
    let clickedXcor mouse-xcor ; updated continuously!
    let clickedYcor mouse-ycor
    ask signs with [ shape = "circletarget" ] [ setxy mouse-xcor mouse-ycor ]
                                     ; pointer is following the mouse
    if mouse-down?
    [
      ask min-one-of foodsources with [ masterpatch? = true ]
        [ distancexy clickedXcor clickedYcor ] ; closest masterpatch is selected
      [
        ifelse member? who SelectedFieldsList = false ; if the masterpatch ID of the
          ; selected field is not already in the list of selected fields..
        [
          ifelse member? "Crop" patchType and length flowerspeciesList = 1 ; and the field
            ; is a crop without a CSO already applied
          [
            set SelectedFieldsList lput who SelectedFieldsList ;..then the masterpatch ID is
              ; added to the list of selected fields
            let memoWho who
            ask foodsources with [ masterpatchID = memoWho ]
            [
              ifelse member? ["plot"] flowerspeciesList
                or member? ["margin"] flowerspeciesList
                or member? ["legume"] flowerspeciesList
              [ set shape "csodeselected" ] ; mark a field to remove the CSO
                [ set shape "csoselected" ] ; mark a selected field
            ]
            wait 0.2
          ]
        ]
        [ user-message "Stewardship options can only be applied to crop fields" ]
      ]
      [ ; if it had been selected already, it is now de-selected!
        set SelectedFieldsList remove who SelectedFieldsList
        let memoWho who
        ask foodsources with [ masterpatchID = memoWho ] [ set shape "circle" ] ; gets back
          ; its original shape
        wait 0.2
      ]
    ]
  ]
  if mouse-inside? = false and timer - startTime > 2 ; user has 2s to move the pointer on map
  [
    ask signs with [ shape = "circletarget" ] [ hide-turtle ]
    if mouse-inside? = false [ set finished? true ] ; if pointer is moved outside,
      ; loop is stopped
  ]
]
end
```

Button 3 "Apply Stewardship"

Calls the procedure *ButtonStewardshipOptionsProc*.

ButtonStewardshipOptionsProc

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

Calls the procedure *CreateHabitatsProc* (see panel "Maps and Settings", button 2) to access the flower data of countryside stewardship options (CSO), saved in the global variables *FlowerSpeciesList_Plot* (*.._Legume, _Margin*). The user is then asked for details of the CSO option, to define their area. Then the stewardship option is applied to all crop fields that have been selected (*SelectedFieldsList*, see button 1 of this panel). CSO can only be applied to agricultural fields, consisting of only a single crop without any CSO already in place. Multiplying the flowers per m2 for each plant species, as specified in *FlowerSpeciesList_...* with the area of the CSO gives the total abundance of these flowers in the field. Dividing this number by the area of the field results in the new flower density, which is saved in the foodsources-own variable *flowerspecies_relativeAbundanceList*. The density of the actual crop is reduced, according to the loss of area due to the application of the CSO. Finally, *StewardshipOptionsUpdateFoodsourcesProc* is called, to create an updated *FoodsourcesFile* (*.._Foodsources.txt*).

to **ButtonStewardshipOptionsProc**

```
; UK Mid-Tier Countryside Stewardship Options: legume: AB15, margin: AB8, plot: AB1
CreateHabitatsProc ; habitat needed to access their data (flower lists/abundances).
                    ; Will die in the subsequent Setup

let marginWidth_m 0
let marginLength_m 0
let plotArea_m2 0
let cssArea_m2 0
let cssFlowerspeciesList []

; get the input for the CSO areas:
if StewardshipOption = "margin"
[
    set marginWidth_m read-from-string user-input "How wide [m] is your margin?"
    set marginLength_m read-from-string user-input "How long [m] is your margin?"
    set cssFlowerspeciesList FlowerSpeciesList_Margin ; flower composition and abundance
                                                         ; in this CSO
]

if StewardshipOption = "plot"
[
    set plotArea_m2 10000 * read-from-string user-input
                                     "Which area [ha] would you like to set aside?"
    set cssFlowerspeciesList FlowerSpeciesList_Plot ; flower composition and abundance
                                                         ; in this CSO
]

if StewardshipOption = "legume"
[ set cssFlowerspeciesList FlowerSpeciesList_Legume ]

; apply CSO to selected fields:
foreach SelectedFieldsList
[
    let cssSpeciesList []
    let cssAbundanceList []
    ; get the flower species & abundances:
    foreach cssFlowerspeciesList ;; goes through all items (item e.g. ["Bugle" 0])
    ; e.g. cssFlowerspeciesList (Plot): ["Alsike_clover" 24.727] ["Bugle" 0]..
    [
        let nextItem read-from-string ? ; e.g. the item ["Common_vetch" 1] is made
                                           ; into the list [Common_vetch 1], i.e. loses the double quotes
        set cssSpeciesList lput item 0 nextItem cssSpeciesList ; e.g. [Alsike_clover
                                                                    Common_knapweed Spear_thistle ..]
        set cssAbundanceList lput item 1 nextItem cssAbundanceList ; e.g. [10 14 10..]
    ]
    ask foodsource ?
    [
        set stewardshipSpeciesList lput StewardshipOption stewardshipSpeciesList
                                     ; StewardshipOption: "margin" "plot" or "legume"
        let i 0
        ; calculate CSO areas:
        if StewardshipOption = "margin" ; area = length x width
        [ set cssArea_m2 marginWidth_m * marginLength_m ]
        if StewardshipOption = "plot" ; area = provided by user
        [ set cssArea_m2 plotArea_m2 ]
    ]
]
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

if StewardshipOption = "legume" ; area = whole field
[ set cssArea_m2 area_sqm ]

if cssArea_m2 > area_sqm ; warning, if CSO area is larger than the actual field!
[
  user-message "Area for stewardship options of one or more fields larger than
               the actual field!"
  set cssArea_m2 area_sqm
]

foreach cssSpeciesList
[
  let currentAbundance item i cssAbundanceList ; this describes the flower density of
          ; margin species in the margin.
          ; However, total patch/field area is larger, hence density must be reduced:
  set currentAbundance precision (currentAbundance * (cssArea_m2 / area_sqm)) 3
  set patchInfo (word cssArea_m2) ; the CSO area is saved as patchInfo
  set stewardshipSpeciesList lput (word "[" ? " " currentAbundance "]")
                                stewardshipSpeciesList

  set i i + 1
]

; now the flower abundance in the actual patch needs to be adjusted/reduced:
let abundanceOld item 1 flowerspecies_relativeAbundanceList
                                ; e.g. [Crop_Field_beans 107.5]
let abundanceNew precision (abundanceOld * (1 - (cssArea_m2 / area_sqm))) 3
set flowerspecies_relativeAbundanceList but-last flowerspecies_relativeAbundanceList
                                ; removes last item, i.e. abundance value
set flowerspecies_relativeAbundanceList lput abundanceNew
                                flowerspecies_relativeAbundanceList ; adds the new abundance value
]
]
StewardshipOptionsUpdateFoodsourcesProc TRUE ; input: userInput? (true)
set SelectedFieldsList []
end

```

StewardshipOptionsUpdateFoodsourcesProc

This procedure creates an updated *FoodsourcesFile* (..*Foodsources.txt*), taking the previously applied countryside stewardship options (CSO) into account. After defining the names of the files, with the option for the user to add a suffix, the map is exported as an image file and then all foodsources are addressed to write their parameter values into the new input file. Then settings are saved (*SaveLoadSettingsProc*) and the *Setup* is called. Finally *foodsources* with a CSO change their shape to visualise the stewardship option that has been added.

```

to StewardshipOptionsUpdateFoodsourcesProc [ userInput? ]
; creates a new FoodsourcesFile "x_Foodsources.txt", in which the new stewardship options
; are taken into account
;; UPDATE THE FOODSOURCES INPUT FILE:
set FoodsourcesFile remove "_SYSTEM_" FoodsourcesFile; "System" files cannot be overwritten
let suffix user-input "Optionally: Provide a suffix for this stewardship scenario
(e.g. 'margins1') to avoid overwriting of your current files - or just click ok: "
set MyMap remove "_SYSTEM_" MyMap
set MyMap remove ".png" MyMap
ifelse suffix = ""
[ set MyMap (word MyMap ".png") ]
[ set MyMap (word MyMap "_" suffix ".png") ]
ask turtles [ hide-turtle ]
export-view MyMap ; exports the view to create an image file of the current map
file-close ; no file should be open right now
set MyParametersFile remove "_SYSTEM_" MyParametersFile
set FoodsourcesFile remove "_SYSTEM_" FoodsourcesFile
set FoodsourcesFile remove "_Foodsources" FoodsourcesFile
set FoodsourcesFile remove ".txt" FoodsourcesFile
ifelse suffix = ""
[ set FoodsourcesFile (word FoodsourcesFile "_Foodsources.txt") ]
[ set FoodsourcesFile (word FoodsourcesFile "_" suffix "_Foodsources.txt") ]

; the old input file is deleted, a new one is created:

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

if file-exists? FoodsourcesFile [ file-delete FoodsourcesFile ]
file-open FoodsourcesFile
file-print 1 / Scaling_NLpatches/m ; equivalent to BS_Scaling_m/NLpatches
file-print count foodsources with [ masterpatch? = true ] ; N food patches
file-print "id patchType patchColour xcor ycor size_sqm flowerSpeciesList perimeter_m info"

; METHOD: first: address all flowerpatches, write the patch data (like size, location etc.)
; in the new file, then address all "layers" of that patch
; layers can occur ("occurrence") in the 'patch' 'margin' 'legume', 'plot'
; (e.g. occurrence = "margin")
foreach sort foodsources with [ masterpatch? = true ]
[
  ask ?
  [
    file-type who file-type " " ; column 1
    file-type "\"""
    file-type patchType ; column 2
    file-type "\"""
    file-type " "
    file-type color + 1 file-type " " ; column 3
    file-type precision xcor 3 file-type " " ; column 4
    file-type precision ycor 3 file-type " " ; column 5
    file-type precision area_sqm 1 file-type " " ; column 6

    let memoWho who
    let flowerSpeciesListNew [] ; this is to re-create flowerspeciesList
    ; but with protected quotation marks
    let occurrenceList fput "patch" CSS_OptionsList ; CSS_OptionsList [ "margin" "legume"
    ; "plot" ]

    let added? false
    foreach sort foodsources with [ masterpatchID = memoWho ] ; all layers of the currently
    ; addressed masterpatch are asked..
    [
      ask ?
      [
        ; occurrenceList = [patch margin legume plot ]
        foreach occurrenceList ; .. first asking those originally in the patch and
        ; then ordered by the CSO option: occurrenceList = [patch margin legume plot]
        [
          if any? foodsources with [ masterpatchID = memoWho and occurrence = ?
          and occurrence != "patch" ] ; adds 'CSO-marker':
          ; "legume" "plot" or "margin" to the list to specify that the following flowerspecies
          ; are not part of the original field but of the CSO
          [
            if added? = false ; make sure it is added only once!
            [ set flowerSpeciesListNew lput (word "[" "\""" ? "\""" "]")
            flowerSpeciesListNew ]
            set added? true
          ]
        ]

        if occurrence = ? ; adds the actual flower species and their abundances:
        [
          set flowerSpeciesListNew lput (word "[" "\"""
          (item 0 flowerspecies_relativeAbundanceList) "\""" " "
          (item 1 flowerspecies_relativeAbundanceList) "]" )
          flowerSpeciesListNew
          let i 0
          foreach stewardshipSpeciesList ; stewardshipSpeciesList = [] or
          ; e.g. [plot [Common_knapweed 0.166] [Greater_knapweed 0.166] ...]
          [
            ifelse member? ? CSS_OptionsList
            [
              set flowerSpeciesListNew lput (word "[" "\""" ? "\""" "]") flowerSpeciesListNew
              ; this adds e.g. "plot" to flowerSpeciesListNew to indicate that the
              ; following flower species are present in the CSO (plot)
              ; flowerSpeciesListNew e.g. ["[" "Crop_Oilseed_rape\" 556.763]" "\"[" "plot\""]"]
              set i i + 1
            ]
            [ ; adding protected quotation marks:
              if length flowerSpeciesList = 1 ; to make sure CSO/blueberry patches
              ; are not repeatedly added
              [
                let newSpeciesData item i stewardshipSpeciesList
                ; e.g. [MarginBirdsfoot_trefoil 1000]
              ]
            ]
          ]
        ]
      ]
    ]
  ]
]

```


Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

        let blankPosition position " " newSpeciesData
        let newSpecies substring newSpeciesData 1 blankPosition
        ; position 0 is "[" hence start at position 1
        let newSpeciesAbundance substring newSpeciesData blankPosition
        (length newSpeciesData - 1) ; last position is "]" hence -1
        set flowerSpeciesListNew lput (word "[" "\" newSpecies "\" " "
            newSpeciesAbundance ")") flowerSpeciesListNew
        set i i + 1
    ]
]
] ; end if occurrence
] ; end foreach occurrenceList
] ; end ask ?
] ; end "foreach sort foodsources"
if length flowerSpeciesList > 1 and member? "Crop" patchType
    and member? who SelectedFieldsList
    [ set flowerSpeciesListNew
        (word "[" "\" (item 0 flowerspecies_relativeAbundanceList)
            "\" " " (item 1 flowerspecies_relativeAbundanceList) "]" ) ]
    file-type flowerSpeciesListNew file-type " " ; ; column 7: flowerSpeciesList
    file-type perimeter_m file-type " " ; ; column 8
    set patchInfo remove "\" patchInfo ; to avoid accumulation of \
    file-write (word patchInfo) ; column 8 ; "no info" ; printed as string i.e. with " "
    file-print ( " ")
]
]
file-close

SaveLoadSettingsProc "Save!" (word "_" suffix) ; update parameters file
Setup

if StewardshipOption = "margin"
[
    ask foodsources with [ shape = "fieldmargin" ] ; field marked "M"
    [
        let memoMasterpatch masterpatchID
        ask foodsources with [ masterpatchID = memoMasterpatch] [ set shape "fieldmargin" ]
    ]
]
if StewardshipOption = "plot"
[
    ask foodsources with [ shape = "fieldplot" ] ; field marked "P"
    [
        let memoMasterpatch masterpatchID
        ask foodsources with [ masterpatchID = memoMasterpatch] [ set shape "fieldplot" ]
    ]
]
if StewardshipOption = "legume"
[
    ask foodsources with [ shape = "fieldlegume" ] ; field marked "L"
    [
        let memoMasterpatch masterpatchID
        ask foodsources with [ masterpatchID = memoMasterpatch] [ set shape "fieldlegume" ]
    ]
]
end

```

Button 4 "Show Stewardship Areas"

Calls the procedure *StewardshipAreasProc*.

StewardshipAreasProc

Asks the user for the total farm area and then displays the area for all applied stewardship options on the output window.

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
to StewardshipAreasProc
  let farmArea_ha read-from-string user-input "What is your total farm area (hectare)?"
  output-print "Total stewardship area: "
  output-print (word precision CSS_TotalAreaAll_ha 2 " ha (" precision (100 *
    CSS_TotalAreaAll_ha / farmArea_ha) 1 " % of farm area)")
  output-print ""
  output-print "Margins: "
  output-print (word precision CSS_TotalAreaMargin_ha 2 " ha (" precision (100 *
    CSS_TotalAreaMargin_ha / farmArea_ha) 1 " % of farm area)")
  output-print ""
  output-print "Legume fields: "
  output-print (word precision CSS_TotalAreaLegume_ha 2 " ha (" precision (100 *
    CSS_TotalAreaLegume_ha / farmArea_ha) 1 " % of farm area)")
  output-print ""
  output-print "Plots: "
  output-print (word precision CSS_TotalAreaPlot_ha 2 " ha (" precision (100 *
    CSS_TotalAreaPlot_ha / farmArea_ha) 1 " % of farm area)")
  output-print ""
end
```

Button 5 "Unselect all Fields"

De-selects all selected fields.

(*PanelButtonProc*):

```
if myButtonCommand = "Unselect all Fields"
[
  foreach SelectedFieldsList ; a list with the ID of all selected masterpatches
  [
    ask foodsources with [ masterpatchID = ? ]
    [ set shape "circle" ]
  ]
  set SelectedFieldsList []
]
```

Button 6 "Define Crop Rotation"

Calls the procedure *CropRotationSelectFilesProc*.

CropRotationSelectFilesProc

Without any rotations, there is only one input "..._foodsources.txt" file linked to a map. This procedure allows the user to define a number input files, which are listed in *CropRotationList*. The user has to define the additional number of input files and then select each of them.

```
to CropRotationSelectFilesProc
  let nRotations read-from-string user-input "Insert number of ADDITIONAL foodsource files you
    would you like to use (insert 0 to remove rotations): "

  let rot 1
  set CropRotationList []
  if nRotations > 0
  [
    set FoodsourcesFile (word remove "_SYSTEM_" FoodsourcesFile)
    set MyMap (word remove "_SYSTEM_" MyMap)
    set CropRotationList lput FoodsourcesFile CropRotationList
  ]
  repeat nRotations
  [
    user-message "Choose a '_Foodsources' text file (this step might be repeated)"
    let nextFile FilenameREP true ; true: file extension will be reported
    set CropRotationList lput nextFile CropRotationList
  ]
end
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

]
set CropRotationListAsString "" ; strings better than lists in parameter file
foreach CropRotationList [ set CropRotationListAsString
                        (word CropRotationListAsString " " ?)]
if user=yes-or-no? "Save settings now?" [ SaveLoadSettingsProc "Save!" "" ]
end

```

CropRotationProc

CropRotationProc is not called by button 6, but by *UpdateMorning_Proc* on 31. December (*Day* = 365) if crop rotations have been defined (*CropRotationList* > 1). It defines the new food sources input file (*FoodsourcesFile*) for the next item in *CropRotationList*. All *foodsources* are removed (*die*) and food source related *bee* and *colony* parameters are deleted. New *foodsources* are then created (*CreateFoodsourcesProc*). Finally, if the procedure was called while running an automated report (see below, Button 7 "Generate My Report!") the file of the generated report is opened.

```

to CropRotationProc
let year ceiling (ticks / 365)
let nMaps length CropRotationList
let mapToUse remainder year nMaps
set FoodsourcesFile item mapToUse CropRotationList
ask foodsources [ die ]
ask bees
[
  set knownMasterpatchesNectarList []
  set knownMasterpatchesPollenList []
  set nectarSourceToGoTo -1
  set pollenSourceToGoTo -1
  set currentFoodsource -1
]
ask colonies ; there shouldn't be any colonies around anyway
[
  set masterpatchesInRangeList []
  set masterpatchesWithNectarlayersInFlowerAndRangeList []
  set masterpatchesWithPollenlayersInFlowerAndRangeList []
  set nectarInFlowerAndRangeList []
  set pollenInFlowerAndRangeList []
]
CreateFoodsourcesProc
ask species
[
  ; create list of foodSources as nest sites and calculate their total area
  set nestSiteFoodsourceList FoodSources with
    [ (member? patchtype [nestHabitatsList] of myself) AND masterPatch? ]
  set nestSiteArea sum [area_sqm] of nestSiteFoodsourceList
  ; queen may start egg laying once 50% of pollen needed to raise 1 batch of eggs is stored:
  set minPollenStore_g 0.5 * 0.001 * devWeightPupationMin_mg * batchsize
    / pollenToBodymassFactor
  if count nestSiteFoodsourceList = 0 and name != "Psithyrus"
    [output-print (word name " has no suitable nesting habitat")]
]

if Report_name != 0 ;; if Report_name != 0, CropRotationProc is called while running
  ;; running the automated report (ButtonGenerateOutputProc)
[ file-open ( word Report_name ".csv" ) ]
end

```

Button 7 "Generate My Report!"

Calls the procedure *ButtonGenerateOutputProc*.

ButtonGenerateOutputProc

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

Runs an analysis of the current map to predict the number of bees supported. The user can choose between two analyses: a "Full simulation", running the model repeatedly (*replicates*) for *timesteps* days. Results are written in a "csv" file. Alternatively, a quick estimation based on the available nectar and pollen resources in the landscapes can be shown.

to ButtonGenerateOutputProc

```

set Report_name ""
let replicates 20
let timesteps 365 * 5
let reportType user-one-of "Choose whether to run a full simulation (may take several hours)
    or just a quick estimate on colony numbers?" ["Full simulation" "Resources provided"]
if reportType = "Resources provided"
[
    let totalPollenOnMapPerYear_kg 0 ; sums up pollen of all foodsources over 1 year
    let totalNectarOnMapPerYear_l 0 ; sums up nectar of all foodsources over 1 year
    ask foodsources
    [
        if stopDay < startDay [ user-message "Foodsource stops flowering before
            it has even started!" ]
        let pollenAdded_kg pollenMax_g * (stopDay - startDay) / 1000
        let nectarAdded_l nectarMax_myl * (stopDay - startDay) / (1000 * 1000)
        set totalPollenOnMapPerYear_kg totalPollenOnMapPerYear_kg + pollenAdded_kg
        set totalNectarOnMapPerYear_l totalNectarOnMapPerYear_l + nectarAdded_l
    ]

    ; Rotheray et al 2017, J Apic Res, Vol. 56, No. 3, 288-299:
    ; colonies require 176 g pollen and 1186 g sugar in their lifetime
    let nColonies_pollen_Rotheray totalPollenOnMapPerYear_kg * 1000 / 176
    let totalSugar_kg totalNectarOnMapPerYear_l * 0.342 ; assuming sucrose concentration of
        ; 1mol/l and 0.342 kg sucrose per mol
    let nColonies_nectar_Rotheray totalSugar_kg / 1.186
    ifelse nColonies_pollen_Rotheray > nColonies_nectar_Rotheray
    [ user-message (word "Total amount of sugar per year: " round totalSugar_kg
        "kg. Total amount of pollen per year: " round totalPollenOnMapPerYear_kg
        "kg. The bee population is limited by lack of nectar! The maximal number of
        colonies supported is up to ca. " round nColonies_nectar_Rotheray) ]
    [ user-message (word "Total amount of sugar per year: " round totalSugar_kg
        "kg. Total amount of pollen per year: " round totalPollenOnMapPerYear_kg
        "kg. The bee population is limited by lack of pollen! The maximal number of
        colonies supported is up to ca. " round nColonies_pollen_Rotheray) ]
]

if reportType = "Full simulation"
[
    set Report_name (word "_" user-input "What is the name of your results?")
    let reportNameSave Report_name ; Report_name needs to be a global variable to be
        ; accessible in CreateFoodsourcesProc, but also local, to not be deleted by Setup!
    if file-exists? ( word Report_name ".csv") [ file-delete ( word Report_name ".csv") ]
    user-message (word "Your results named " Report_name " are about to be created - this may
        take a while! To speed things up, set the 'slider' to faster and untick
        the 'view updates' box. The progress of the simulation is shown in the
        'Output' window")
    file-open ( word Report_name ".csv")
    file-type "Timestep ," file-type "Colonies ," file-type "Pollinators ,"
    file-type remove ".csv" Report_name file-type " ," file-type replicates
    file-type " ," file-type timesteps file-print " ,"
    set RAND_SEED 1 ; random seed is set, to make the results reproducible
    repeat replicates
    [
        Setup
        set Report_name reportNameSave ; needs to be re-set after "clear-all"
        output-print (word "Progress of the simulation: " precision (100 * (RAND_SEED /
            replicates)) 1 " %")

        file-open ( word Report_name ".csv")
        repeat timesteps
        [
            Go
            file-type ticks file-type " ," file-type TotalColonies file-type " ,"
            file-print TotalAdultworkers
            if AssertionViolated = true
            [
                ask patches [ set pcolor red ]
            ]
        ]
    ]
]

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

        stop
      ]
    ]
    file-close
    set RAND_SEED RAND_SEED + 1 ; new set of pseude-random numbers
  ]
  user-message (word "Your results named " Report_name " are now finished! Create a report
    using the My BEE-STEWARD Report.xlsm file")
]
end

```

Panel "Display Options"

This panel provides additional information for the user, e.g. showing nectar or pollen visits on the map, parameter values or input files currently in use.

Button 1 "Default view"

Calls the procedure *ButtonDisplayProc* with "defaultView" as input.

ButtonDisplayProc

Shows the default view of the map. The procedure requires one input (*displayOption*), which can be "nectarVisits", "pollenVisits" or "defaultView" to specify, what is shown on the map. The local variable *averageCumulVisits* sums up *cumulNectarVisits* (or *cumulPollenVisits*) of each *foodsource* and is then divided by the number of food patches ("masterpatches"). *CumulNectarVisits* (*cumulPollenVisits*) is set to 0 each 1. January and continuously increased by the number of bee visits at this specific *foodsource* ("layer").

```

to ButtonDisplayProc [ displayOption ] ; nectarVisits pollenVisits defaultView
  let averageCumulVisits 0 ; sums up (nectar OR pollen) visits of ALL foodsources and is
                                ; finally divided by N foodsources

  let displayColor red
  if displayOption = "nectarVisits" or displayOption = "pollenVisits"
  [
    ask foodsources
    [
      show-turtle
      if displayOption = "nectarVisits"
      [
        set averageCumulVisits averageCumulVisits + cumulNectarVisits
        set displayColor yellow ; nectar visits shown in yellow
      ]
      if displayOption = "pollenVisits"
      [
        set averageCumulVisits averageCumulVisits + cumulPollenVisits
        set displayColor orange ; pollen visits shown in orange
      ]
    ]
  ]
  set averageCumulVisits averageCumulVisits / count foodsources with [masterpatch? = true]
  ask foodsources with [ masterpatch? = true ]
  [
    let myMasterpatch who
    let summedCumulVisits 0 ; all pollen or nectar visits at THIS food patch
    ask foodsources with [ masterpatchID = myMasterpatch ] ; addressing all "layers" of
                                                                ; the current food patch

    [
      if displayOption = "nectarVisits"
        [ set summedCumulVisits summedCumulVisits + cumulNectarVisits ]
      if displayOption = "pollenVisits"
        [ set summedCumulVisits summedCumulVisits + cumulPollenVisits ]
    ]
  ]

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

]
let displayVisits summedCumulVisits / averageCumulVisits ; visits here relative to mean
ask foodsources with [ masterpatchID = myMasterpatch ] ; all foodsources of current food
; patch change their colour, to reflect the relative number of visits at the patch
[ ; color depends on relative number of visits to this food patch:
  set color scale-color displayColor sqrt displayVisits 0 4 ; scale-color: 4 inputs:
                                ; color number range1 range2
  ifelse round (displayVisits * averageCumulVisits) > 1 ; # visits shown, if >= 0.5
    [ set label-color 103 set label round (displayVisits * averageCumulVisits) ]
    [ set label-color 103 set label "" ] ; no label shown, if patch wasn't visited
  ]
]
]
if displayOption = "defaultView" ; default display of the map
[
  ask deadCols [ set color white set size 1 set label "" ]
  ask foodsources [ set color colorMemo ]
  ask colonies [ set label colonySize ]
  ifelse showFoodSources?
  [
    ifelse showMasterPatchesOnly?
    [
      ask Foodsources with [masterPatch?] [show-turtle set label "" ]
      ask Foodsources with [not masterPatch?] [hide-turtle]
    ]
    [ ask Foodsources [st set label "" ] ]
  ]
  [ ask foodsources [ht] ]

  ifelse showSearchingQueens?
  [ ask bees with [ caste = "queen" and colonyID = -1 and activity != "hibernate" ] [st] ]
  [ ask bees with [ caste = "queen" and colonyID = -1 and activity != "hibernate" ] [ht] ]

  ifelse showCohorts?
  [ ask bees with [ shape = "halflife" ] [st] ]
  [ ask bees with [ shape = "halflife" ] [ht] ]

  ifelse showQueens?
  [ ask bees with [ caste = "queen" and mated? = true ] [st] ]
  [ ask bees with [ caste = "queen" and mated? = true ] [ht] ]

  ifelse showNests?
  [ ask colonies [st] ]
  [ ask colonies [ht] ]

  ifelse showDeadCols?
  [ ask deadCols [st] ]
  [ ask deadCols [ht] ]

  ask badgers [ st]

  ifelse showGrid?
  [
    ask patches with [ remainder pxcor round (gridsize * Scaling_NLpatches/m) = 0 ]
    [ set pcolor black ]
    ask patches with [ remainder pycor round (gridsize * Scaling_NLpatches/m) = 0 ]
    [ set pcolor black ]
    ask patch 290 5 [ set plabel-color black set plabel word gridsize " m" ]
  ]
  [
    ask patches [ set pcolor pcolorSave ]
    ask patch 290 5 [ set plabel "" ]
  ]
]
end

```

Button 2 "Show Nectar Visits"

Calls the procedure *ButtonDisplayProc* (see above) with "nectarVisits" as input.

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

Button 3 "Show Pollen Visits"

Calls the procedure *ButtonDisplayProc* (see above) with "pollenVisits" as input.

Button 4 "Identify!"

Calls the procedure *ButtonIdentifyProc*.

ButtonIdentifyProc

Allows the user to identify elements shown on the map by hovering the mouse pointer above them. An "identifier" *sign* is created and shown while the mouse pointer is within the map. Uses the label of the *sign* to display information about the agent it is hovering over.

```
to ButtonIdentifyProc
  let finished? false
  let displayedInfo ""
  let startTime timer
  ask foodsources with [ masterpatch? = false ] [ hide-turtle ]
  create-signs 1
  [
    set shape "identifier"
    set size 20
    set color black
  ]
  while [ finished? = false ] ; continuously looped, while mouse within map
  [
    let hoveredXcor mouse-xcor ; current position of mouse
    let hoveredYcor mouse-ycor
    let turtleID -1
    ask signs with [ shape = "identifier" ] ; sign follows mouse
    [ setxy mouse-xcor mouse-ycor show-turtle ]
    ask min-one-of turtles with [ shape != "identifier" ] ; closest agent is identified..
    [distancexy hoveredXcor hoveredYcor]
    [
      ifelse distancexy hoveredXcor hoveredYcor < 5 ; ..and info is shown, if distance < 5
      [
        set turtleID who ; exact info shown depends on the breed of the closest agent
        if breed = foodsources [ set displayedInfo patchType ]
        if breed = colonies [ set displayedInfo (word shape " colony") ]
        if breed = bees [ set displayedInfo (word speciesName ": " caste " " stage) ]
        if breed = signs [ set displayedInfo shape ]
      ]
      [
        set displayedInfo ""
        set turtleID -1
      ]
    ]
  ]
  ask signs with [ shape = "identifier" ] [ set label displayedInfo ]
  if mouse-down? = true ; detailed info, if user clicks on agent:
  [
    ifelse turtleID >= 0
    [ inspect turtle turtleID ] ; opens "inspect" window of selected turtle
    [ inspect patch mouse-xcor mouse-ycor ]
  ]
  wait 0.2
  if mouse-inside? = false and timer - startTime > 2 ; ends while loop
  [
    ask signs with [ shape = "identifier" ] [ hide-turtle ]
    let outsideTime timer
    let stopOutsideCheck false
    while [ stopOutsideCheck = false ]
    [
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
wait 0.5
if mouse-inside? = true [ set stopOutsideCheck true ]
if timer - outsideTime > 2
[
  set stopOutsideCheck true
  set finished? true
]
]
ask signs with [ shape = "identifier" ] [ show-turtle ]
]
ask signs with [ shape = "identifier" ] [ die ] ; identifier sign is deleted
end
```

Button 5 "Show Input Files"

Shows the names of all input files in the output window.

```
if myButtonCommand = "Show Input Files"
[
  output-type "Input file: " output-print FoodsourcesFile
  output-type "Parameter file: " output-print MyParametersFile
  output-type "Habitat file: " output-print HabitatsFile
  output-type "Flower species file: " output-print FlowerspeciesFile
  output-type "Bumblebee species file: " output-print BeespeciesFile
  output-type "TextMap file: " output-print TextMap
]
```

Button 6 " Show Parameter Values"

Lists the parameters defined in the parameter input file and show their current values in the output window.

```
if myButtonCommand = "Show Parameter Values"
[
  foreach AllParametersList
  [
    let parameter remove " " ? ; remove possible blanks
    let command (word "set GenericRunCommandValue " parameter)
    run command
    output-print (word parameter ": " GenericRunCommandValue)
  ]
]
```

Button 7 " More Display Options"

Calls the procedure *ButtonDisplayButtonsProc*.

ButtonDisplayButtonsProc

Provides the information to replace the map by a virtual panel with additional display options. Then calls the procedure *VirtualButtonsProc*.

```
to ButtonDisplayButtonsProc
let buttonLabelsAndCommandsList
[
  ["DISPLAY OPTIONS" "" ]
  [ "ShowCohorts" "set ShowCohorts? " ]
  [ "ShowDeadCols" "set ShowDeadCols? " ]
]
```


Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
[ "ShowFoodsources" "set ShowFoodsources? " ]
[ "ShowGrid" "set ShowGrid? " ]
[ "ShowInspectedColony" "set ShowInspectedColony? " ]
[ "ShowMasterpatchesOnly" "set ShowMasterpatchesOnly? " ]
[ "ShowNests" "set ShowNests? " ]
[ "ShowPlots" "set ShowPlots? " ]
[ "ShowQueens" "set ShowQueens? " ]
[ "ShowSearchingQueens" "set ShowSearchingQueens? " ]
[ "ShowWeather" "set ShowWeather? " ]
]
VirtualButtonsProc buttonLabelsAndCommandsList
end
```

VirtualButtonsProc

Based on the information (*buttonLabelsAndCommandsList*) from *ButtonDisplayButtonsProc*, the tick boxes and labels are displayed. For the labels, the actual agent (turtle) is invisible, only its label is displayed. Tick boxes can switch between two shapes, "switchButtonOn" and "switchButtonOff", depending whether they checked or not. Also, three virtual buttons are shown: "Apply and Save" to apply the changes made and save them (*SaveLoadSettingsProc*) in the project's parameter file, "Apply" to apply the changes without saving them and "Cancel" to discard any changes. Finally, the world returns to the map view (*UpdateViewProc*).

```
to VirtualButtonsProc [ buttonLabelsAndCommandsList ]
; buttonLabelsAndCommandsList: [{"DISPLAY OPTIONS" ""} ["ShowCohorts" "set ShowCohorts? "]]...
ask patches [ set pcolor 9 ] ; whitish background
ask turtles [ hide-turtle ] ; hide all turtles
let buttonXinit min-pxcor + 60 ; positions of labels and tick boxes
let buttonYinit max-pycor - 16
let buttonX buttonXinit
let buttonY buttonYinit
let labelXshift -10
let labelYshift -2
let buttonsYdistance 8
let buttonsXdistance 20
let headerShift_x 20
let leave? false
let save? false
let apply? false

foreach buttonLabelsAndCommandsList
[
  create-buttons 1 ; creates a tick box
  [
    setxy buttonX buttonY
    set size 7
    set color 7
    set myLabel item 0 ?
    set myCommand item 1 ?
    set exitButton? false
    set headerButton? false
    if myCommand = ""
    [ set headerButton? true
      hide-turtle ]
    set on? BS_SwitchValueButtonREP myCommand ; SwitchValueButtonREP => true or false
    ifelse on? = true
    [ set shape "switchButtonOn" ] ; shape depends on the current value (true/false)
    [ set shape "switchButtonOff" ]
  ]
  create-buttonLabels 1 ; the LABEL (actual turtle is invisible, only its label is shown)
  [
    setxy buttonX + labelXshift buttonY + labelYshift
    set shape "invisible"
    set color grey
    set size 0.1
    set label-color black
    set label item 0 ?
  ]
]
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
        if item 1 ? = ""
        [ set xcor xcor + headerShift_x ]
    ]
    set buttonY buttonY - buttonsYdistance
    if buttonY < 1.5 * buttonsYdistance
    [
        set buttonX buttonX + labelXshift
        set buttonY buttonYinit
    ]
] ; end foreach

create-buttons 1 ; the APPLY & SAVE BUTTON
[
    setxy max-pxcor - 20 max-pycor - 20
    set shape "circle"
    set size 12
    set color green
    set myLabel "APPLY & SAVE"
    set exitButton? true
    set on? false
]

create-buttonLabels 1 ; the APPLY & SAVE LABEL
[
    setxy (max-pxcor - 20 + labelXshift) (max-pycor - 20 + labelYshift)
    set shape "invisible"
    set color grey
    set size 0.1
    set label-color black
    set label "APPLY & SAVE"
]

create-buttons 1 ; the APPLY BUTTON
[
    setxy max-pxcor - 20 max-pycor - 40
    set shape "circle"
    set size 12
    set color green
    set myLabel "APPLY"
    set exitButton? true
    set on? false
]

create-buttonLabels 1 ; the APPLY LABEL
[
    setxy (max-pxcor - 20 + labelXshift) (max-pycor - 40 + labelYshift)
    set shape "invisible"
    set color grey
    set size 0.1
    set label-color black
    set label "APPLY"
]

create-buttons 1 ; the CANCEL BUTTON
[
    setxy max-pxcor - 20 max-pycor - 60
    set shape "circle"
    set size 12
    set color red
    set myLabel "CANCEL"
    set exitButton? true
    set on? false
]

create-buttonLabels 1 ; the CANCEL LABEL
[
    setxy (max-pxcor - 20 + labelXshift) (max-pycor - 60 + labelYshift)
    set shape "invisible"
    set color grey
    set size 0.1
    set label-color black
    set label "CANCEL"
]

while [ leave? = false ]
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
[
  if mouse-down?
  [
    let chosenButton nobody
    ask patch mouse-xcor mouse-ycor
    [
      if count buttons with [ distance myself < (buttonsYdistance / 2) ] > 0
      [ set chosenButton [who] of one-of buttons ; closest button is chosen
        with [ distance myself < (buttonsYdistance / 2) ] ]
    ]
    if chosenButton != nobody
    [
      ask button chosenButton
      [
        ifelse on? = false
        [ set on? true set shape "switchButtonOn" ] ; tick
        [ set on? false set shape "switchButtonOff" ] ; untick
      ]
      display
      wait 0.2
    ]
  ]

  ask buttons with [ myLabel = "APPLY & SAVE" ]
  [
    if on? = true
    [
      set leave? true
      set save? true ; includes "apply"
    ]
  ]

  ask buttons with [ myLabel = "APPLY" ]
  [
    if on? = true
    [
      set leave? true
      set apply? true
    ]
  ]

  ask buttons with [ myLabel = "CANCEL" ]
  [
    if on? = true
    [
      set leave? true
      set save? false
    ]
  ]
] ; while leave? false

if save? = true or apply? = true
[
  let commandList []
  foreach sort-on [ who ] buttons
  [ ask ?
    [
      ifelse on? = true
      [
        if myCommand != "" and myCommand != 0
        [
          set myCommand (word myCommand "true")
          set commandList lput myCommand commandList
        ]
      ]
      [
        if myCommand != "" and myCommand != 0
        [
          set myCommand (word myCommand "false")
          set commandList lput myCommand commandList
        ]
      ]
    ]
  ]
]
foreach commandList [ run ? ]
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```

]
ask buttons [ die ] ; buttons & labels are no longer needed
ask buttonLabels [ die ]
UpdateViewProc
if save? = true [ SaveLoadSettingsProc "Save!" "" ]
end

```

SwitchValueButtonREP

Reports true or false, depending on the current value of the parameter defined in the input (*command*).

```

to-report BS_SwitchValueButtonREP [ command ]
  set command remove "set " command
  if command = "" [ set command "false" ]
  set command (word "set SwitchOn " command) ; ; SwitchOn is set to the value of the reported
                                              ; parameter (true or false)
  run command
  report SwitchOn
end

```

UpdateViewProc

This procedure shows or hides agents ("turtles"), depending on the current setting.

```

to UpdateViewProc
  ask deadCols [ set color white set size 1 set label "" ]
  ask foodsources [ set color colorMemo ]
  ask Colonies [ set label ColonySize ]
  ifelse showFoodSources?
  [
    ifelse showMasterPatchesOnly?
    [
      ask Foodsources with [masterPatch?] [show-turtle]
      ask Foodsources with [not masterPatch?] [hide-turtle]
    ]
    [ ask Foodsources [show-turtle]]
  ]
  [ ask foodsources [hide-turtle]]
  ifelse showSearchingQueens?
  [ ask bees with [ caste = "queen" and colonyID = -1 and activity != "hibernate" ]
    [show-turtle]]
  [ ask bees with [ caste = "queen" and colonyID = -1 and activity != "hibernate" ]
    [hide-turtle]]
  ifelse showCohorts?
  [ ask bees with [ shape = "halflin" ] [show-turtle]]
  [ ask bees with [ shape = "halflin" ] [hide-turtle]]
  ifelse showQueens?
  [ ask bees with [ caste = "queen" and mated? = true ] [show-turtle]]
  [ ask bees with [ caste = "queen" and mated? = true ] [hide-turtle]]
  ifelse showNests?
  [ ask colonies [show-turtle]]
  [ ask colonies [hide-turtle]]
  ifelse showDeadCols?
  [ ask deadCols [show-turtle]]
  [ ask deadCols [hide-turtle]]
  ask badgers [ show-turtle]
  let labelPatch patch 295 5
  ifelse showGrid?
  [
    ask patches [ set pcolor pcolorSave ]
    ask patches with [ remainder pxcor round (gridsize * Scaling_NLpatches/m) = 0 ]
    [ set pcolor black ]
    ask patches with [ remainder pycor round (gridsize * Scaling_NLpatches/m) = 0 ]
    [ set pcolor black ]
  ]

```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
ask labelPatch [ set plabel-color black set plabel word gridsize " m" ]
]
[
ask patches [ set pcolor pcolorSave ]
ask labelPatch [ set plabel "" ]
]
end
```

Panel "Advanced Input Options"

This panel provides additional information for the user, e.g. showing nectar or pollen visits on the map, parameter values or input files currently in use.

Button 1 "Set Random Seed"

Allows the user to set *RAND_SEED*, which initialises NetLogo's pseudo-random number generator. If it is set to 0, the pseudo-random number generator is not initialised.

```
if member? "Set Random Seed " myButtonCommand
[
set RAND_SEED read-from-string user-input "New value for RAND_SEED
(if 0: random-seed is not set): "
set Button1Monitor (word "Set Random Seed (" RAND_SEED ")")
]
```

Button 2 "Advanced Setup Options"

Calls the procedure *ButtonAdvancedSetupOptionsProc*.

ButtonAdvancedSetupOptionsProc

This procedure provides the information to set up a virtual panel for advanced setup options (similar to *ButtonDisplayButtonsProc*, see above). It then calls *VirtualButtonsProc* to display the virtual panel (see panel "Display Options", button 7).

```
to ButtonAdvancedSetupOptionsProc
let buttonLabelsAndCommandsList
[
["ADVANCED SETUP OPTIONS" "" ]
[ "KeepDeadColonies" "set KeepDeadColonies? " ]
[ "MergeHedges" "set MergeHedges? " ]
[ "MinSizeFoodSources" "set MinSizeFoodSources? " ]
[ "RemoveEmptyFoodSources" "set RemoveEmptyFoodSources? " ]
[ "SexLocus" "set SexLocus? " ]
[ "StopExtinct" "set StopExtinct? " ]
[ "UnlimitedMales" "set UnlimitedMales? " ]
]
VirtualButtonsProc buttonLabelsAndCommandsList
end
```

Button 3 "Add a Background Image"

Calls the procedure *ButtonBackgroundImageProc*.

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

ButtonBackgroundImageProc

Replaces the map image by an image chosen by the user (e.g. a satellite photo of the area).

```
to ButtonBackgroundImageProc
  user-message "Select an image file to be used as background image"
  set BackgroundImage user-file
  import-pcolors BackgroundImage
  ask patches [ set satelliteColor pcolor ]
end
```

Button 7 "VERSION TEST"

Calls the procedure *VersionTestProc*.

VersionTestProc

Sets up the model in a defined way and runs it for two years. The result is compared to an expected result in order to determine whether or not the model or one of the input files have been changed.

```
to VersionTestProc ; to test whether the model was changed
  let expectedValue 46703
  set GenericPlot1 "Number of colonies for different species" ; not really needs setting, as
                                                                ; output runs with "local randomness"

  set MyMap "_SYSTEM_Sussex1.png"
  set MyParametersFile "_SYSTEM_Parameters.csv"
  set ProjectsOwnParameterFile? false
  ClearProc
  set RAND_SEED 1
  random-seed RAND_SEED
  if ProjectsOwnParameterFile? = true
  [
    set MyParametersFile (word remove "_SYSTEM_" MyMap) ; the parameter file of this project
                                                                ; has the same name as the map used, but make sure, system files cannot be overwritten
    set MyParametersFile (word remove ".png" MyParametersFile) ; MyMap is an image file
                                                                ; but MyParametersFile is not - delete extension!
    set MyParametersFile (word MyParametersFile "_Parameters.csv") ; identifier for parameter
                                                                ;file and correct extension
  ]
  PanelSettingProc
  ReadAllParametersProc
  set BeeSpeciesInitialQueensList [ ["B_hortorum" 20] ["B_hypnorum" 20] ["B_lapidarius" 20]
                                     ["B_pascuorum" 20] ["B_pratorum" 20] ["B_terrestris" 100] ["Psithyrus" 20] ]
  set N_Badgers 5
  ParametersProc
  CreateFoodsourcesProc
  CreateSpeciesProc
  CreateBadgersProc
  CreateInitialQueensProc
  UpdateMorning_Proc
  CreateSignsProc
  OutputDailyProc
  if ShowGrid? = true
  [
    ask patches with [ remainder pxcor round (Gridsize * Scaling_NLpatches/m) = 0 ]
    [ set pcolor black ]
    ask patches with [ remainder pycor round (Gridsize * Scaling_NLpatches/m) = 0 ]
    [ set pcolor black ]
    ask patch 290 5 [ set plabel-color black set plabel word Gridsize " m" ]
  ]
  repeat 2 * 365
  [
    Go
    if AssertionViolated = true
    [
```

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.

```
ask patches [ set pcolor red ]
stop
]
]
let testValue TotalBeesEverProduced + TotalHibernatingQueens + TotalMales
+ TotalAdultWorkers + TotalFoodSources

ifelse testValue = expectedValue
[ user-message (word "No deviation detected from the official Beesteward (2020) version
                    (Test value: " expectedValue ")") ]

[
  type testValue type "      Difference: " print testValue - expectedValue
  user-message "CHANGES MADE TO THE MODEL OR INPUT FILES!"
]
end
```

INPUT FILES

- **Parameter file** (*MyParametersFile*, e.g. "_SYSTEM_Parameters.csv"): a new input file, which mainly defines parameters that have previously been placed on the the user interface of *Bumble-BEEHAVE* and *BEESCOUT*.

- **Image map** (*MyMap*, e.g. "_SYSTEM_Example_Farm.png"): equivalent to *InputMap* in *Bumble-BEEHAVE* and *InputFile* (if set to an image file) in *BEESCOUT*. Ideal dimensions are (multiples of) 300 x 210 pixels. Alternatively, the map can also be created from a text file.

- **Text map** (e.g. "_SYSTEM_Textmap_Sussex1.txt"): equivalent to *InputFile* (if set to "TextMap") in *BEESCOUT*. Creating a new map from a text file will automatically also create an image map, linked to this project. Please note that the format of the file has changed in comparison to *Bumble-BEEHAVE*.

- **Food sources** (*FoodsourcesFile*, e.g. "_SYSTEM_Example_Farm_Foodsources.txt"): as *INPUT_FILE* in *Bumble-BEEHAVE*, however, the columns "quantityPollen_g", "proteinPollenProp", "quantityNectar_l", "concentration_mol/l", "startDay", "stopDay", "corollaDepth_mm", "nectarFlowerVolume_myl" and "intFlowerTime_s" have been removed, as these data are provided by the flower species input file. Instead, "Perimeter_m" has been added, describing the perimeter of the food source.

- **Flower species** (*FlowerspeciesFile*, e.g. "_SYSTEM_Flowerspecies.csv"): identical to *FlowerspeciesFile* in *Bumble-BEEHAVE*.

- **Habitats** (*HabitatsFile*, e.g. "_SYSTEM_Habitats.csv"): as *HabitatsInput* in *BEESCOUT*, however, the first column "Colour" was replaced by the columns "habitatColourID", "colourName_forUserInfoOnly", "colourRangeMin", "colourRangeMax" and "habitatSwitchedOn?", defining the range of colours associated with a particular habitat type and whether or not this habitat will be applied. "colourName_forUserInfoOnly" describes the habitat colour in words (using NetLogo terminology). However, this information is not read in by the model, does not affect the simulation and only serves to inform the user. Furthermore, an additional column ("ONLY FLOWER DATA BEYOND THIS COLUMN!") separates the habitat data from the lists of flower abundancies.

- **Bee species** (*BeespeciesFile*, "_SYSTEM_BumbleSpecies_UK_01.csv"): identical to *SpeciesFilename* in *Bumble-BEEHAVE*.

Twiston-Davies, Becher & Osborne. (2021). BEE-STEWARD: a research and decision support software for effective land management to promote bumblebee populations. *Methods in Ecology and Evolution*.