

```

1 | #include "fabm_driver.h"
2 | #include "fabm_0d.h"
3 | !-----
4 | !BOP
5 | !
6 | !MODULE: Output manager
7 | !
8 | !INTERFACE:
9 |   module output
10 | !
11 | !DESCRIPTION:
12 | ! TODO
13 | !
14 |   ! From FABM
15 |   use fabm
16 |   use fabm_types
17 |   use fabm_driver
18 |   use fabm_properties
19 |
20 |   ! From GOTM
21 |   use time
22 |   use field_manager
23 |   use output_manager_core, only:output_manager_host=>host, type_output_manager_host=>type_host
24 |   use output_manager
25 |
26 |   use shared
27 |
28 |   implicit none
29 |
30 |   private
31 |
32 |   public configure_output, init_output, do_output, clean_output
33 |
34 |   public register_output_fields, fm
35 |
36 |   type, extends(type_output_manager_host) :: type_fabm0d_host
37 |   contains
38 |     procedure :: julian_day    => fabm0d_host_julian_day
39 |     procedure :: calendar_date => fabm0d_host_calendar_date
40 |   end type
41 |
42 |   type (type_field_manager), target :: fm
43 |
44 |   integer, parameter :: ASCII_FMT = 1
45 |   integer, parameter :: NETCDF_FMT = 2
46 |
47 |   character(len=PATH_MAX) :: output_file
48 |   integer, public :: output_format
49 |   logical :: add_environment
50 |   logical :: add_conserved_quantities
51 |   logical :: add_diagnostic_variables
52 |   integer(timestepkind) :: nsave
53 |
54 |   integer :: out_unit = -1
55 |
56 |   character, parameter :: separator = char(9)
57 | !EOP
58 | !-----
59 |
60 |   contains
61 |
62 | !-----
63 | !BOP
64 | ! !IROUTINE: configure output from namelists or YAML
65 | !
66 | !INTERFACE:
67 |   subroutine configure_output(namlst)
68 | !
69 | !DESCRIPTION:
70 | ! TODO
71 | !
72 | !INPUT PARAMETERS:
73 |   integer, intent(in) :: namlst
74 | !
75 | !REVISION HISTORY:
76 | ! Original author(s): Karsten Bolding
77 | !
78 | !LOCAL PARAMETERS:
79 |   namelist /output/ output_file,output_format,nsave,add_environment, &
80 |     add_diagnostic_variables, add_conserved_quantities
81 |   integer :: ios
82 | !
83 | !EOP
84 | !-----
85 | !BOC
86 |   ! Read output namelist
87 |   output_file = ''
88 |   output_format = ASCII_FMT
89 |   nsave = 1
90 |   add_environment = .false.
91 |   add_conserved_quantities = .false.
92 |   add_diagnostic_variables = .false.
93 |
94 |   read(namlst,nml=output,iostat=ios)
95 |   if (ios/=0) call driver%fatal_error('configure_output','run.nml: I could not read the "output" namelist.')
96 |   if (output_file==='') call driver%fatal_error('configure_output','run.nml: "output_file" must be set to a valid file
97 | path in "output" namelist.')

```

```

98 | end subroutine configure_output
99 |EOC
100 |
101 |-----
102 |BOP
103 | !IROUTINE: prepare for output
104 | !
105 | !INTERFACE:
106 |   subroutine init_output(start)
107 | !
108 | !DESCRIPTION:
109 |   TODO
110 | !
111 | !INPUT PARAMETERS:
112 |   character(len=*) , intent(in) :: start
113 | !
114 | !REVISION HISTORY:
115 |   Original author(s): Karsten Bolding
116 | !
117 | !LOCAL PARAMETERS:
118 |   integer           :: i
119 |   real(rk),pointer :: pdata
120 |EOP
121 |-----
122 |BOC
123 |   do i=1,size(model%interior_state_variables)
124 |     pdata => model%get_data(model%get_interior_variable_id(model%interior_state_variables(i)%name))
125 |     call fm%send_data(model%interior_state_variables(i)%name, pdata)
126 |   end do
127 |   do i=1,size(model%bottom_state_variables)
128 |     pdata => model%get_data(model%get_horizontal_variable_id(model%bottom_state_variables(i)%name))
129 |     call fm%send_data(model%bottom_state_variables(i)%name, pdata)
130 |   end do
131 |   do i=1,size(model%surface_state_variables)
132 |     pdata => model%get_data(model%get_horizontal_variable_id(model%surface_state_variables(i)%name))
133 |     call fm%send_data(model%surface_state_variables(i)%name, pdata)
134 |   end do
135 |
136 |   do i=1,size(model%interior_diagnostic_variables)
137 |     if (model%interior_diagnostic_variables(i)%save) then
138 |       pdata => model%get_interior_diagnostic_data(i)
139 |       call fm%send_data(model%interior_diagnostic_variables(i)%name, pdata)
140 |     end if
141 |   end do
142 |   do i=1,size(model%horizontal_diagnostic_variables)
143 |     if (model%horizontal_diagnostic_variables(i)%save) then
144 |       pdata => model%get_horizontal_diagnostic_data(i)
145 |       call fm%send_data(model%horizontal_diagnostic_variables(i)%name, pdata)
146 |     end if
147 |   end do
148 |
149 | end subroutine init_output
150 |EOC
151 |
152 |-----
153 |BOP
154 | !IROUTINE: do the output
155 | !
156 | !INTERFACE:
157 |   subroutine do_output(n)
158 | !
159 | !DESCRIPTION:
160 |   TODO
161 | !
162 | !INPUT PARAMETERS:
163 |   integer(timestepkind) , intent(in) :: n
164 | !
165 | !REVISION HISTORY:
166 |   Original author(s): Karsten Bolding
167 | !
168 |EOP
169 |-----
170 |BOC
171 |   call output_manager_save(julianday,secondsofday,int(n))
172 |
173 | end subroutine do_output
174 |EOC
175 |
176 |-----
177 |BOP
178 | !
179 | !IROUTINE: Clean up.
180 | !
181 | !INTERFACE:
182 |   subroutine clean_output(ignore_errors)
183 | !
184 | !DESCRIPTION:
185 |   Close all open files.
186 | !
187 | !INPUT PARAMETERS:
188 |   logical , intent(in) :: ignore_errors
189 | !
190 | !REVISION HISTORY:
191 |   Original author(s): Jorn Bruggeman
192 | !
193 | !LOCAL PARAMETERS:
194 |   integer :: iret
195 |EOP

```

```

196 !-----
197 !BOC
198
199   if (out_unit/=1) close(out_unit)
200
201   call output_manager_clean()
202   call fm%finalize()
203
204   end subroutine clean_output
205 !EOC
206
207   subroutine register_output_fields()
208
209     integer :: i
210     logical :: in_output
211
212     LEVEL1 'output_manager'
213     allocate(type_fabm@d_host::output_manager_host)
214     call output_manager_init(fm,title)
215
216     LEVEL1 'field_manager'
217     call fm%register_dimension('lon',1,id=id_dim_lon)
218     call fm%register_dimension('lat',1,id=id_dim_lat)
219     call fm%register_dimension('time',id=id_dim_time)
220     call fm%initialize(prepend_by_default=(/id_dim_lon,id_dim_lat/),append_by_default=(/id_dim_time/))
221     call fm%register('lon','degrees_east','longitude',dimensions=(/id_dim_lon/),no_default_dimensions=.true.,data@d=lon
222     gitude,coordinate_dimension=id_dim_lon)
223     call fm%register('lat','degrees_north','latitude',dimensions=(/id_dim_lat/),no_default_dimensions=.true.,data@d=lat
224     itude,coordinate_dimension=id_dim_lat)
225
226     call fm%register('par','W/m^2','par',standard_name='downwelling_photosynthetic_radiative_flux',data@d=par)
227     call fm%register('temp','Celsius','temperature',standard_name='sea_water_temperature',data@d=temp%value)
228     call fm%register('salt','1e-3','salinity',standard_name='sea_water_practical_salinity',data@d=salt%value)
229
230     ! state variables
231     do i=1,size(model%interior_state_variables)
232       in_output = register(model%interior_state_variables(i))
233     end do
234     do i=1,size(model%bottom_state_variables)
235       in_output = register(model%bottom_state_variables(i))
236     end do
237     do i=1,size(model%surface_state_variables)
238       in_output = register(model%surface_state_variables(i))
239     end do
240
241     ! diagnostic variables
242     do i=1,size(model%interior_diagnostic_variables)
243       model%interior_diagnostic_variables(i)%save = register(model%interior_diagnostic_variables(i))
244     end do
245     do i=1,size(model%horizontal_diagnostic_variables)
246       model%horizontal_diagnostic_variables(i)%save = register(model%horizontal_diagnostic_variables(i))
247     end do
248
249     ! conserved quantities
250     do i=1,size(model%conserved_quantities)
251       call fm%register('int_change_in_'//trim(model%conserved_quantities(i)%name), trim(model%conserved_quantities(i)%
252       units)//'*m', 'integrated change in '//trim(model%conserved_quantities(i)%long_name), &
253       minimum=model%conserved_quantities(i)%minimum, maximum=model%conserved_quantities(i)%maximum,
254       fill_value=model%conserved_quantities(i)%missing_value, &
255       category='fabm/conservation', output_level=output_level_default, used=in_output, data@d=int_ch
256       ange_in_totals(i))
257       if (in_output) compute_conserved_quantities = .true.
258     end do
259
260     contains
261
262     function register(variable) result(used)
263       class (type_fabm_variable), intent(in) :: variable
264       logical :: used
265
266       integer :: output_level
267       type (type_field), pointer :: field
268       class (type_property), pointer :: property
269
270       output_level = output_level_default
271       if (variable%output==output_none) output_level = output_level_debug
272       call fm%register(variable%name, variable%units, variable%long_name, &
273       minimum=variable%minimum, maximum=variable%maximum, fill_value=variable%missing_value, &
274       category='fabm'//variable%target%owner%get_path(), output_level=output_level, used=used, field=
275       field)
276       property => variable%properties%first
277       do while (associated(property))
278         select type (property)
279           class is (type_real_property)
280             call field%set_attribute(property%name,property%value)
281         end select
282         property => property%next
283       end do
284     end function register
285
286   end subroutine register_output_fields
287
288   subroutine fabm@d_host_julian_day(self,yyyy,mm,dd,julian)
289     class (type_fabm@d_host), intent(in) :: self
290     integer, intent(in) :: yyyy,mm,dd
291     integer, intent(out) :: julian
292     call julian_day(yyyy,mm,dd,julian)
293   end subroutine

```

```
288 |
289 | subroutine fabm0d_host_calendar_date(self,julian,yyyy,mm,dd)
290 |   class (type_fabm0d_host), intent(in) :: self
291 |   integer, intent(in) :: julian
292 |   integer, intent(out) :: yyyy,mm,dd
293 |   call calendar_date(julian,yyyy,mm,dd)
294 | end subroutine
295 |
296 | -----
297 |
298 | end module output
299 |
300 | -----
301 | Copyright Bolding & Bruggeman ApS - GNU Public License - www.gnu.org
302 | -----
```