**Activity Diagram (new)**

**https://plantuml.com/edit.png**

Old [syntax for activity diagram](https://plantuml.com/activity-diagram-legacy) had several limitations and drawbacks (for example, it's difficult to maintain).

So a completely new syntax and implementation is now available to users. Another advantage of this implementation is that it's done without the need of having Graphviz installed (as for sequence diagrams).

This syntax will replace the old legacy one. However, for compatibility reason, the old syntax will still be recognized, to ensure *ascending compatibility*.

Users are simply encouraged to migrate to the new syntax.

**https://plantuml.com/backtop3.pngSimple action**

**https://plantuml.com/edit.png**

Activities label starts with : and ends with ;.

Text formatting can be done using [creole wiki syntax](https://plantuml.com/creole).

They are implicitly linked in their definition order.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  :Hello world;  :This is defined on  several \*\*lines\*\*;  @enduml  https://plantuml.com/imgw/img-c3867e9cac0cf6256ce7f78538f7a37e.png |

**https://plantuml.com/backtop3.pngStart/Stop/End**

**https://plantuml.com/edit.png**

You can use start and stop keywords to denote the beginning and the end of a diagram.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  :Hello world;  :This is defined on  several \*\*lines\*\*;  stop  @enduml  https://plantuml.com/imgw/img-cc3abb09d78f82efd7f8243d24c213d4.png |

You can also use the end keyword.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  :Hello world;  :This is defined on  several \*\*lines\*\*;  end  @enduml  https://plantuml.com/imgw/img-6483f342aa96c925d6c4c21ec038e07d.png |

**https://plantuml.com/backtop3.pngConditional**

**https://plantuml.com/edit.png**

You can use if, then and else keywords to put tests in your diagram. Labels can be provided using parentheses.

The 3 syntaxes are possible:

* if (...) then (...)

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  if (Graphviz installed?) then (yes)  :process all\ndiagrams;  else (no)  :process only  \_\_sequence\_\_ and \_\_activity\_\_ diagrams;  endif  stop  @enduml  https://plantuml.com/imgw/img-9bf949080ad116a57cd375f94965521d.png |

* if (...) is (...) then

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  if (color?) is (<color:red>red) then  :print red;  else  :print not red;  @enduml  https://plantuml.com/imgw/img-0799482cc962673e5f6fd32a8d58e306.png |

* if (...) equals (...) then

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  if (counter?) equals (5) then  :print 5;  else  :print not 5;  @enduml  https://plantuml.com/imgw/img-6febbfa48135297a4a0ad79d567c0da8.png |

*[Ref.*[*QA-301*](https://forum.plantuml.net/301/activity-diagram-beta?show=302#a302)*]*

**Several tests (horizontal mode)**

You can use the elseif keyword to have several tests *(by default, it is the horizontal mode)*:

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  if (condition A) then (yes)  :Text 1;  elseif (condition B) then (yes)  :Text 2;  stop  (no) elseif (condition C) then (yes)  :Text 3;  (no) elseif (condition D) then (yes)  :Text 4;  else (nothing)  :Text else;  endif  stop  @enduml  https://plantuml.com/imgw/img-be390c1085aad318ff13583fc1706640.png |

**Several tests (vertical mode)**

You can use the command !pragma useVerticalIf on to have the tests in vertical mode:

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  !pragma useVerticalIf on  start  if (condition A) then (yes)  :Text 1;  elseif (condition B) then (yes)  :Text 2;  stop  elseif (condition C) then (yes)  :Text 3;  elseif (condition D) then (yes)  :Text 4;  else (nothing)  :Text else;  endif  stop  @enduml  https://plantuml.com/imgw/img-756c7fad21ac604edc9d746de7e5532b.png |

You can use the -P[command-line](https://plantuml.com/command-line) option to specify the pragma:

java -jar plantuml.jar -PuseVerticalIf=on

*[Refs.*[*QA-3931*](https://forum.plantuml.net/3931/please-provide-elseif-structure-vertically-activity-diagrams)*,*[*issue-582*](https://github.com/plantuml/plantuml/issues/582)*]*

**https://plantuml.com/backtop3.pngSwitch and case [switch, case, endswitch]**

**https://plantuml.com/edit.png**

You can use switch, case and endswitch keywords to put switch in your diagram.

Labels can be provided using parentheses.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  switch (test?)  case ( condition A )  :Text 1;  case ( condition B )  :Text 2;  case ( condition C )  :Text 3;  case ( condition D )  :Text 4;  case ( condition E )  :Text 5;  endswitch  stop  @enduml  https://plantuml.com/imgw/img-e7fd3a78ac677a4dd4e2290db650b61a.png |

**https://plantuml.com/backtop3.pngConditional with stop on an action [kill, detach]**

**https://plantuml.com/edit.png**

You can stop action on a if loop.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  if (condition?) then  :error;  stop  endif  #palegreen:action;  @enduml  https://plantuml.com/imgw/img-99385f324a7719bffdc85c66796c02fc.png |

But if you want to stop at the precise action, you can use the kill or detach keyword:

* kill

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  if (condition?) then  #pink:error;  kill  endif  #palegreen:action;  @enduml  https://plantuml.com/imgw/img-48c7a9048ad59fa6af50f10ce41d6abb.png |

*[Ref.*[*QA-265*](https://forum.plantuml.net/265/new-activity-diagram-syntax-direction-of-links?show=306#a306)*]*

* detach

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  if (condition?) then  #pink:error;  detach  endif  #palegreen:action;  @enduml  https://plantuml.com/imgw/img-da3b0b0c22104912fc0f1ae2ced6fbbf.png |

**https://plantuml.com/backtop3.pngRepeat loop**

**https://plantuml.com/edit.png**

You can use repeat and repeatwhile keywords to have repeat loops.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  repeat  :read data;  :generate diagrams;  repeat while (more data?) is (yes)  ->no;  stop  @enduml  https://plantuml.com/imgw/img-cfadf84e26c51a7003ecfa4220c85c71.png |

It is also possible to use a full action as repeat target and insert an action in the return path using the backward keyword.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  repeat :foo as starting label;  :read data;  :generate diagrams;  backward:This is backward;  repeat while (more data?)  stop  @enduml  https://plantuml.com/imgw/img-d9800579042a54c5bcfe85687cb9a8f1.png |

**https://plantuml.com/backtop3.pngBreak on a repeat loop [break]**

**https://plantuml.com/edit.png**

You can use the break keyword after an action on a loop.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  repeat  :Test something;  if (Something went wrong?) then (no)  #palegreen:OK;  break  endif  ->NOK;  :Alert "Error with long text";  repeat while (Something went wrong with long text?) is (yes) not (no)  ->//merged step//;  :Alert "Success";  stop  @enduml  https://plantuml.com/imgw/img-ec49b4ac1f7220090acac8419bc2dc35.png |

*[Ref.*[*QA-6105*](https://forum.plantuml.net/6105/possible-to-draw-a-line-to-another-box-via-id-or-label?show=6107#a6107)*]*

**https://plantuml.com/backtop3.pngWhile loop**

**https://plantuml.com/edit.png**

You can use while and endwhile keywords to have repeat loops.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  while (data available?)  :read data;  :generate diagrams;  endwhile  stop  @enduml  https://plantuml.com/imgw/img-bf13ed29bafb8e01a3fba39dd555e258.png |

It is possible to provide a label after the endwhile keyword, or using the is keyword.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  while (check filesize ?) is (not empty)  :read file;  endwhile (empty)  :close file;  @enduml  https://plantuml.com/imgw/img-751b40b5f1ea77d5398282b961fdc7f2.png |

If you are using detach to form an infinite while loop, then you will want to also hide the partial arrow that results using -[hidden]->

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  :Step 1;  if (condition1) then  while (loop forever)  :Step 2;  endwhile  -[hidden]->  detach  else  :end normally;  stop  endif  @enduml  https://plantuml.com/imgw/img-0aa6a6025fb2402a3a8b0c9b0387a200.png |

**https://plantuml.com/backtop3.pngParallel processing [fork, fork again, end fork, end merge]**

**https://plantuml.com/edit.png**

You can use fork, fork again and end fork or end merge keywords to denote parallel processing.

**Simple fork**

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  fork  :action 1;  fork again  :action 2;  end fork  stop  @enduml  https://plantuml.com/imgw/img-02129d0ba38d986c015ae7ad84ba8053.png |

**fork with end merge**

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  fork  :action 1;  fork again  :action 2;  end merge  stop  @enduml  https://plantuml.com/imgw/img-e6593b86f8833f64f5abaaa56b897ce6.png |

*[Ref.*[*QA-5320*](https://forum.plantuml.net/5320/please-provide-fork-without-join-with-merge-activity-diagrams?show=5321#a5321)*]*

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  fork  :action 1;  fork again  :action 2;  fork again  :action 3;  fork again  :action 4;  end merge  stop  @enduml  https://plantuml.com/imgw/img-bf1fae37d077f988cb440a1e7e9f85a6.png |

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  fork  :action 1;  fork again  :action 2;  end  end merge  stop  @enduml  https://plantuml.com/imgw/img-8c34ab608e971c8d772bcdbc9f1b28a1.png |

*[Ref.*[*QA-13731*](https://forum.plantuml.net/13731)*]*

**Label on end fork (or UML joinspec):**

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  fork  :action A;  fork again  :action B;  end fork {or}  stop  @enduml  https://plantuml.com/imgw/img-c062bfdf451522bf6e997ea317dbdf66.png |

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  fork  :action A;  fork again  :action B;  end fork {and}  stop  @enduml  https://plantuml.com/imgw/img-9e9803aef488302ad2e90143c82de66f.png |

*[Ref.*[*QA-5346*](https://forum.plantuml.net/5346/please-inplement-joinspec-for-join-nodes?show=5348#a5348)*]*

**Other example**

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  if (multiprocessor?) then (yes)  fork  :Treatment 1;  fork again  :Treatment 2;  end fork  else (monoproc)  :Treatment 1;  :Treatment 2;  endif  @enduml  https://plantuml.com/imgw/img-e46a0063b1cae05aa465856bdab53984.png |

**https://plantuml.com/backtop3.pngSplit processing**

**https://plantuml.com/edit.png**

**Split**

You can use split, split again and end split keywords to denote split processing.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  split  :A;  split again  :B;  split again  :C;  split again  :a;  :b;  end split  :D;  end  @enduml  https://plantuml.com/imgw/img-09f1d4cab3e643308e26501fee577588.png |

**Input split (multi-start)**

You can use hidden arrows to make an input split (multi-start):

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  split  -[hidden]->  :A;  split again  -[hidden]->  :B;  split again  -[hidden]->  :C;  end split  :D;  @enduml  https://plantuml.com/imgw/img-af68992417dd44cfb7b9bd745073e6c9.png |

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  split  -[hidden]->  :A;  split again  -[hidden]->  :a;  :b;  split again  -[hidden]->  (Z)  end split  :D;  @enduml  https://plantuml.com/imgw/img-718d588b341f399ddf79a76068f999e0.png |

*[Ref.*[*QA-8662*](https://forum.plantuml.net/8662)*]*

**Output split (multi-end)**

You can use kill or detach to make an output split (multi-end):

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  split  :A;  kill  split again  :B;  detach  split again  :C;  kill  end split  @enduml  https://plantuml.com/imgw/img-a79c68bfdf86b083b332cb0df4853a4a.png |

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  split  :A;  kill  split again  :b;  :c;  detach  split again  (Z)  detach  split again  end  split again  stop  end split  @enduml  https://plantuml.com/imgw/img-f926bfbf4ef00aac8d826cdded8ca4d3.png |

**https://plantuml.com/backtop3.pngNotes**

**https://plantuml.com/edit.png**

Text formatting can be done using [creole wiki syntax](https://plantuml.com/creole).

A note can be floating, using floating keyword.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  :foo1;  floating note left: This is a note  :foo2;  note right  This note is on several  //lines// and can  contain <b>HTML</b>  ====  \* Calling the method ""foo()"" is prohibited  end note  stop  @enduml  https://plantuml.com/imgw/img-839b2ea669ad4144a8ee7b819a128a9c.png |

You can add note on backward activity:

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  repeat :Enter data;  :Submit;  backward :Warning;  note right: Note  repeat while (Valid?) is (No) not (Yes)  stop  @enduml  https://plantuml.com/imgw/img-91d957c9b83a5bb904b45b428401b52f.png |

*[Ref.*[*QA-11788*](https://forum.plantuml.net/11788/is-it-possible-to-add-a-note-to-backward-activity?show=11802#a11802)*]*

You can add note on partition activity:

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  partition "\*\*process\*\* HelloWorld" {  note  This is my note  ----  //Creole test//  end note  :Ready;  :HelloWorld(i)>  :Hello-Sent;  }  @enduml  https://plantuml.com/imgw/img-2ae6bb4d475241bebe449ee8800af0d7.png |

*[Ref.*[*QA-2398*](https://forum.plantuml.net/2398/is-it-possible-to-add-a-comment-on-top-of-a-activity-partition?show=2403#a2403)*]*

**https://plantuml.com/backtop3.pngColors**

**https://plantuml.com/edit.png**

You can specify a [color](https://plantuml.com/color) for some activities.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  :starting progress;  #HotPink:reading configuration files  These files should be edited at this point!;  #AAAAAA:ending of the process;  @enduml  https://plantuml.com/imgw/img-2e4af8a52df9d28c8818e2a34b57e1d0.png |

You can also use [gradient color](https://plantuml.com/color).

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  partition #red/white testPartition {  #blue\green:testActivity;  }  @enduml  https://plantuml.com/imgw/img-920443a53b5659510ac52c15762bbfec.png |

*[Ref.*[*QA-4906*](https://forum.plantuml.net/4906/setting-ad-hoc-gradient-backgrounds-in-activity?show=4917#a4917)*]*

**https://plantuml.com/backtop3.pngLines without arrows**

**https://plantuml.com/edit.png**

You can use skinparam ArrowHeadColor none in order to connect activities using lines only, without arrows.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam ArrowHeadColor none  start  :Hello world;  :This is on defined on  several \*\*lines\*\*;  stop  @enduml  https://plantuml.com/imgw/img-97d5d8ec88ad630ab90a53c0da822b69.png |

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam ArrowHeadColor none  start  repeat :Enter data;  :Submit;  backward :Warning;  repeat while (Valid?) is (No) not (Yes)  stop  @enduml  https://plantuml.com/imgw/img-d9114613c2cf336da7f5ed018461cf22.png |

**https://plantuml.com/backtop3.pngArrows**

**https://plantuml.com/edit.png**

Using the -> notation, you can add texts to arrow, and change their [color](https://plantuml.com/color).

It's also possible to have dotted, dashed, bold or hidden arrows.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  :foo1;  -> You can put text on arrows;  if (test) then  -[#blue]->  :foo2;  -[#green,dashed]-> The text can  also be on several lines  and \*\*very\*\* long...;  :foo3;  else  -[#black,dotted]->  :foo4;  endif  -[#gray,bold]->  :foo5;  @enduml  https://plantuml.com/imgw/img-551f2aade711a3788bbd7bb9c9a2b391.png |

**https://plantuml.com/backtop3.pngConnector**

**https://plantuml.com/edit.png**

You can use parentheses to denote connector.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  :Some activity;  (A)  detach  (A)  :Other activity;  @enduml  https://plantuml.com/imgw/img-fcd46104f99974a4514d8c9eca1e5d9e.png |

**https://plantuml.com/backtop3.pngColor on connector**

**https://plantuml.com/edit.png**

You can add [color](https://plantuml.com/color) on connector.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  :The connector below  wishes he was blue;  #blue:(B)  :This next connector  feels that she would  be better off green;  #green:(G)  stop  @enduml  https://plantuml.com/imgw/img-0f8738089a752aec570e1ee076bb9972.png |

*[Ref.*[*QA-10077*](https://forum.plantuml.net/10077/assigning-color-to-connectors?show=10080#c10080)*]*

**https://plantuml.com/backtop3.pngGrouping or partition**

**https://plantuml.com/edit.png**

**Group**

You can group activity together by defining group:

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  group Initialization  :read config file;  :init internal variable;  end group  group Running group  :wait for user interaction;  :print information;  end group  stop  @enduml  https://plantuml.com/imgw/img-c15698a348c396f0ca03466ae4986d71.png |

**Partition**

You can group activity together by defining partition:

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  partition Initialization {  :read config file;  :init internal variable;  }  partition Running {  :wait for user interaction;  :print information;  }  stop  @enduml  https://plantuml.com/imgw/img-44204842980ccfb84e1335c32a109eb5.png |

It's also possible to change partition [color](https://plantuml.com/color):

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  partition #lightGreen "Input Interface" {  :read config file;  :init internal variable;  }  partition Running {  :wait for user interaction;  :print information;  }  stop  @enduml  https://plantuml.com/imgw/img-8b57e14a78334eec430dd25844de5157.png |

*[Ref.*[*QA-2793*](https://forum.plantuml.net/2793/activity-beta-partition-name-more-than-one-word-does-not-work?show=2798#a2798)*]*

It's also possible to add [link](https://plantuml.com/link) to partition:

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  partition "[[http://plantuml.com partition\_name]]" {  :read doc. on [[http://plantuml.com plantuml\_website]];  :test diagram;  }  end  @enduml  https://plantuml.com/imgw/img-bfd67fc160b9cc58436a137017a5645c.png |

*[Ref.*[*QA-542*](https://forum.plantuml.net/542/ability-to-define-hyperlink-on-diagram-elements?show=14003#c14003)*]*

**Group, Partition, Package, Rectangle or Card**

You can group activity together by defining:

* group;
* partition;
* package;
* rectangle;
* card.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  group Group  :Activity;  end group  floating note: Note on Group  partition Partition {  :Activity;  }  floating note: Note on Partition  package Package {  :Activity;  }  floating note: Note on Package  rectangle Rectangle {  :Activity;  }  floating note: Note on Rectangle  card Card {  :Activity;  }  floating note: Note on Card  end  @enduml  https://plantuml.com/imgw/img-0509c5e931b135b7d25896d658f24f19.png |

**https://plantuml.com/backtop3.pngSwimlanes**

**https://plantuml.com/edit.png**

Using pipe |, you can define swimlanes.

It's also possible to change swimlanes [color](https://plantuml.com/color).

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  |Swimlane1|  start  :foo1;  |#AntiqueWhite|Swimlane2|  :foo2;  :foo3;  |Swimlane1|  :foo4;  |Swimlane2|  :foo5;  stop  @enduml  https://plantuml.com/imgw/img-3823b36e6d0f90d236b5e6604274b3da.png |

You can add if conditional or repeat or while loop within swimlanes.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  |#pink|Actor\_For\_red|  start  if (color?) is (red) then  #pink:\*\*action red\*\*;  :foo1;  else (not red)  |#lightgray|Actor\_For\_no\_red|  #lightgray:\*\*action not red\*\*;  :foo2;  endif  |Next\_Actor|  #lightblue:foo3;  :foo4;  |Final\_Actor|  #palegreen:foo5;  stop  @enduml  https://plantuml.com/imgw/img-f82ef7ac2a686539d79924a89e5f0ff7.png |

You can also use alias with swimlanes, with this syntax:

* |[#<color>|]<swimlane\_alias>| <swimlane\_title>

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  |#palegreen|f| fisherman  |c| cook  |#gold|e| eater  |f|  start  :go fish;  |c|  :fry fish;  |e|  :eat fish;  stop  @enduml  https://plantuml.com/imgw/img-2ec4ac2357e3371a25b2a405f50c461e.png |

*[Ref.*[*QA-2681*](https://forum.plantuml.net/2681/possible-define-alias-swimlane-place-alias-everywhere-else?show=2685#a2685)*]*

**https://plantuml.com/backtop3.pngDetach or kill [detach, kill]**

**https://plantuml.com/edit.png**

It's possible to remove an arrow using the detach or kill keyword:

* detach

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  :start;  fork  :foo1;  :foo2;  fork again  :foo3;  detach  endfork  if (foo4) then  :foo5;  detach  endif  :foo6;  detach  :foo7;  stop  @enduml  https://plantuml.com/imgw/img-313afe52216f271cc13131bec0a19afa.png |

* kill

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  :start;  fork  :foo1;  :foo2;  fork again  :foo3;  kill  endfork  if (foo4) then  :foo5;  kill  endif  :foo6;  kill  :foo7;  stop  @enduml  https://plantuml.com/imgw/img-5069dd009b4eb497f68d50694671b5f2.png |

**https://plantuml.com/backtop3.pngSDL (Specification and Description Language)**

**https://plantuml.com/edit.png**

By changing the final ; separator, you can set different rendering for the activity:

* |
* <
* >
* /
* \\
* ]
* }

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  :Ready;  :next(o)|  :Receiving;  split  :nak(i)<  :ack(o)>  split again  :ack(i)<  :next(o)  on several lines|  :i := i + 1]  :ack(o)>  split again  :err(i)<  :nak(o)>  split again  :foo/  split again  :bar\\  split again  :i > 5}  stop  end split  :finish;  @enduml  https://plantuml.com/imgw/img-5767e6ac655a94e6b2bd388ceb00b5e6.png |

**https://plantuml.com/backtop3.pngComplete example**

**https://plantuml.com/edit.png**

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  :ClickServlet.handleRequest();  :new page;  if (Page.onSecurityCheck) then (true)  :Page.onInit();  if (isForward?) then (no)  :Process controls;  if (continue processing?) then (no)  stop  endif  if (isPost?) then (yes)  :Page.onPost();  else (no)  :Page.onGet();  endif  :Page.onRender();  endif  else (false)  endif  if (do redirect?) then (yes)  :redirect process;  else  if (do forward?) then (yes)  :Forward request;  else (no)  :Render page template;  endif  endif  stop  @enduml  https://plantuml.com/imgw/img-b5dab4490953dacc51e0da7d68063d48.png |

**https://plantuml.com/backtop3.pngCondition Style**

**https://plantuml.com/edit.png**

**Inside style (by default)**

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam conditionStyle inside  start  repeat  :act1;  :act2;  repeatwhile (<b>end)  :act3;  @enduml  https://plantuml.com/imgw/img-8394820b3578cac4cadce4482c94bb65.png |

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  repeat  :act1;  :act2;  repeatwhile (<b>end)  :act3;  @enduml  https://plantuml.com/imgw/img-303fecc921c4aa400fbce22db1b7e5dd.png |

**Diamond style**

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam conditionStyle diamond  start  repeat  :act1;  :act2;  repeatwhile (<b>end)  :act3;  @enduml  https://plantuml.com/imgw/img-6399aa3da4f0aaaceea613217fb49894.png |

**InsideDiamond (or *Foo1*) style**

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam conditionStyle InsideDiamond  start  repeat  :act1;  :act2;  repeatwhile (<b>end)  :act3;  @enduml  https://plantuml.com/imgw/img-e77167cea8cbe2c98b7d0c4fd74215ed.png |

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam conditionStyle foo1  start  repeat  :act1;  :act2;  repeatwhile (<b>end)  :act3;  @enduml  https://plantuml.com/imgw/img-708df926de932f0805fdbd379321d9aa.png |

*[Ref.*[*QA-1290*](https://forum.plantuml.net/1290/plantuml-condition-rendering)*and*[*#400*](https://github.com/plantuml/plantuml/issues/400#issuecomment-721287124)*]*

**https://plantuml.com/backtop3.pngCondition End Style**

**https://plantuml.com/edit.png**

**Diamond style (by default)**

* With one branch

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam ConditionEndStyle diamond  :A;  if (decision) then (yes)  :B1;  else (no)  endif  :C;  @enduml  https://plantuml.com/imgw/img-60b06354d175495a8d7592dba73b2a63.png |

* With two branches (B1, B2)

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam ConditionEndStyle diamond  :A;  if (decision) then (yes)  :B1;  else (no)  :B2;  endif  :C;  @enduml  @enduml  https://plantuml.com/imgw/img-d99c3bb5f306978e6c5a088a7f287edc.png |

**Horizontal line (hline) style**

* With one branch

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam ConditionEndStyle hline  :A;  if (decision) then (yes)  :B1;  else (no)  endif  :C;  @enduml  https://plantuml.com/imgw/img-94197cbc824f97e7866095f3d650e867.png |

* With two branches (B1, B2)

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  skinparam ConditionEndStyle hline  :A;  if (decision) then (yes)  :B1;  else (no)  :B2;  endif  :C;  @enduml  @enduml  https://plantuml.com/imgw/img-a63b53a39818bf6ec86f1bfefac692b5.png |

*[Ref.*[*QA-4015*](https://forum.plantuml.net/4015/its-possible-to-draw-if-else-endif-without-merge-symbol)*]*

**https://plantuml.com/backtop3.pngUsing (global) style**

**https://plantuml.com/edit.png**

**Without style *(by default)***

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  start  :init;  -> test of color;  if (color?) is (<color:red>red) then  :print red;  else  :print not red;  note right: no color  endif  partition End {  :end;  }  -> this is the end;  end  @enduml  https://plantuml.com/imgw/img-73baf777a0447df5cdb8ae00d0879736.png |

**With style**

You can use [style](https://plantuml.com/style-evolution) to change rendering of elements.

|  |  |
| --- | --- |
| https://plantuml.com/clip3.png | @startuml  <style>  activityDiagram {  BackgroundColor #33668E  BorderColor #33668E  FontColor #888  FontName arial  diamond {  BackgroundColor #ccf  LineColor #00FF00  FontColor green  FontName arial  FontSize 15  }  arrow {  FontColor gold  FontName arial  FontSize 15  }  partition {  LineColor red  FontColor green  RoundCorner 10  BackgroundColor PeachPuff  }  note {  FontColor Blue  LineColor Navy  BackgroundColor #ccf  }  }  document {  BackgroundColor transparent  }  </style>  start  :init;  -> test of color;  if (color?) is (<color:red>red) then  :print red;  else  :print not red;  note right: no color  endif  partition End {  :end;  }  -> this is the end;  end  @enduml  https://plantuml.com/imgw/img-38fd7dca9c07a7b5afd552f564702c0d.png |