Version: 4.0.1

How to use multiple XML files

In the examples we have presented, we always create an entire Tree and its SubTrees from a **single XML file**.

But as the number of SubTrees grows, it is convenient to use multiple files.

Our subtrees

File **subtree_A.xml**:

```
<root>
    <BehaviorTree ID="SubTreeA">
        <SaySomething message="Executing Sub_A" />
    </BehaviorTree>
</root>
```

File **subtree_B.xml**:

```
<root>
    <BehaviorTree ID="SubTreeB">
        <SaySomething message="Executing Sub_B" />
    </BehaviorTree>
</root>
```

Load multiple files manually (recommended)

Let's consider a file **main_tree.xml** that should include the other 2 files:

```
<root>
    <BehaviorTree ID="MainTree">
        <Sequence>
            <SaySomething message="starting MainTree" />
            <SubTree ID="SubTreeA" />
            <SubTree ID="SubTreeB" />
        </Sequence>
    </BehaviorTree>
<root>
```

To load multiple files manually:

```
int main()
{
  BT::BehaviorTreeFactory factory;
  factory.registerNodeType<DummyNodes::SaySomething>("SaySomething");

  // Find all the XML files in a folder and register all of them.
  // We will use std::filesystem::directory_iterator
  std::string search_directory = "./";

  using std::filesystem::directory_iterator;
  for (auto const& entry : directory_iterator(search_directory))
  {
    if( entry.path().extension() == ".xml")
    {
      factory.registerBehaviorTreeFromFile(entry.path().string());
    }
  }
  // This, in our specific case, would be equivalent to
  // factory.registerBehaviorTreeFromFile("./main_tree.xml");
  // factory.registerBehaviorTreeFromFile("./subtree_A.xml");
  // factory.registerBehaviorTreeFromFile("./subtree_B.xml");

  // You can create the MainTree and the subtrees will be added automatically.
  std::cout << "----- MainTree tick ----" << std::endl;
  auto main_tree = factory.createTree("MainTree");
  main_tree.tickWhileRunning();

  // ... or you can create only one of the subtrees
  std::cout << "----- SubA tick ----" << std::endl;
  auto subA_tree = factory.createTree("SubTreeA");
  subA_tree.tickWhileRunning();

  return 0;
```

```
}
/* Expected output:

Registered BehaviorTrees:
 - MainTree
 - SubTreeA
 - SubTreeB
----- MainTree tick ----
Robot says: starting MainTree
Robot says: Executing Sub_A
Robot says: Executing Sub_B
----- SubA tick ----
Robot says: Executing Sub_A
```

Add multiple files with "include"

If you prefer to move the information of the trees to include into the XML itself, you can modify **main_tree.xml** as shown below:

```
<root BTCPP_format="4">
    <include path="./subtree_A.xml" />
    <include path="./subtree_B.xml" />
    <BehaviorTree ID="MainTree">
        <Sequence>
            <SaySomething message="starting MainTree" />
            <SubTree ID="SubTreeA" />
            <SubTree ID="SubTreeB" />
        </Sequence>
    </BehaviorTree>
<root>
```

As you may notice, we included two relative paths in **main_tree.xml** that tells `BehaviorTreeFactory` where to find the required dependencies.

Paths are relative to **main_tree.xml**.

We can now create the tree as usual:

```
factory.createTreeFromFile("main_tree.xml")
```

Edit this page