CODING CONVENTIONS

Don't hesitate to let other dev review your code if you have any doubts about any aspect of your code.

NAMING

- Pointers name should be prefixed by a **p.**
  -> char* **p**Test = NULL
- Inputs parameters should be prefixed by a **i.**
  -> void foo(char * **i**pTest);
- Output parameters should be prefixed by **o.**
  -> void bar(char *& **o**pTest);
- Private class members should be prefixed by a **_**.
  -> class MyClass { char * _pMyPrivateMember; };
- If non trivial loop, give **non-obscure name to iterators**.

```
-> for ( int i_graph =1; i_graph <= n_graphs; ++i_graph)
        for (int i_arc = 1; i_arc <= n_arcs; ++i_arc)
                // doStuff
```

- **Camel case** preferred.
  -> int thisIsMyVeryOwnInteger;

General practices

- **Catch errors** early (even if no backup method is implemented - just print something).
  - Errors shall be named with the className upperCases + an error numbers starting at 0000.

```
-> Exemple :
In the Class Engine :
        std::cerr << "E000 : Failed to Init the Engine." << std::endl;
        Std::cerr << "E001: Failed to create the Player entity." << std::endl;
In the class TextureManager :
        Std::cerr << "TM0000 : Failed to init the TextureManager." << std::endl;
        Std::cerr << "TM001 : Failed to init to load the texture ." << std::endl;
```

- Use C++11 features like **smart pointers** as much as possible over raw pointers.
- Avoid **friend** classes .
- Use **Preprocessor directives** to enclose header files as mush as possible.

```
-> In MyClass.h :
 #ifndef MyClass_H
#define MyClass_H
// All the stuff
#endif
```

- Avoid **static data members** unless it is justified and safe.
  - Example : Instance counters
- Avoid **implicit** type cast.
- Use **const** whenever it's possible.
- Try to pass **void** to every function with **empty parameters**.
  - Void foo(void) { // doStuff }
- Apply the **"Rule of Three"** as much as possible.