Media Engine
Code Commenting Format

This documents shall describe the standard *code commenting* format that should be used on our Media Engine code.

- To add description for some *class, class method, struct, enum or union,* use C-style comment block starting with “**/****” and ends with “***/**” like below:

```
/**

Your class, method or structure description here

*/
```

Doxygen also supports some others variations for *comment block*. For simplicity and consistency, they are not recommended to use in media.

- To add single line comment on class/struct/enum *variables*, use “///” to start. For example

```
/// Represents whether acoustic echo canceller is enabled or not

bool m_bEchoEnabled;
```

Inside the block comment describing some method, the following doxygen commands must be used when appropriate.

- **@param**: used to start describing a parameter. The syntax is as below

@param <*[in] / [out] / [in,out]*> <*parameter_name*> “Your parameter description”

Here [in] / [out] / [in,out] specifies the direction of the parameter, i.e. [in] means the parameter is an input to the method, [in,out] means the parameter is used both input to the method and provides output with the same parameter. Must be used when passing pointers, addresses and so where the usage is ambiguous.

Note that, you may also provide description for several parameters using a single @param command as below:

```
/** Sets the position.

* @param x,y,z Coordinates of the position in 3D space.

*/
```

```
void setPosition(double x,double y,double z,double t)

{

}
```

- **@return**: Specifies what is returned by the method. Syntax is

@return "Description of return value"

Note that, for void types, this is not required.

- **@throws:** If the method throws some exception, then you must mention it on the description using throws. Below is the syntax

@throws <exception_object_name> "Exception Description"

for example

```
@throws IoError Thrown if `fileName` could not be written to.
```

- **@tparam**: Used usually one for each template parameter. Each tag should have a description following the parameter name. You do *not* usually need to document default values. Syntax is as below

@tparam <template_parameter_name> "Parameter description"

- **@notes**/**@warnings:** *Notes and warnings* are *optional sections* that provides additional information about the code, possibly including a discussion of the algorithm or known limitations of the code. Syntax is

@note "Text"

@warning "Warning message"

- **@deprecated**: Should use this tag, where applicable, to warn users that the component is deprecated. The text that follows the @deprecated tag should include:

    1. In what version the object was deprecated, and when it will be removed.
    2. Reason for deprecation if this is useful information (object is superseded, duplicates functionality found elsewhere, etc.)
    3. New recommended way of obtaining the same functionality.

The syntax is

@deprecated "Description"