**How to use the game engine**

In order to use the game engine, you need to make your main class extend CoreUser. You must then override the init() and tick(double delta) methods.

In your main method, write the following:

CoreSystem core = new CoreSystem(new [Insert the name of your main class here](), [Insert the intended display title here], [Insert the intended display width here], [Insert the intended display height here]);

core.start();

For example:

CoreSystem core = new CoreSystem(new Test(), “GameEngine Test”, 640, 480);

core.start();

The above code would create an empty display window titled “GameEngine Test” and of width 640 and height 480.

**Settings**

After typing:

CoreSystem core = new CoreSystem(new [Insert the name of your main class here](), [Insert the intended display title here], [Insert the intended display width here], [Insert the intended display height here]);

But before calling the start() method, you can set some settings for the application.

|  |  |
| --- | --- |
| setPrintFramesPerSecond(boolean printFramesPerSecond) | Enable/disable whether the program prints the FPS to the console. (Enabled by default). |
| setPrintVersionData(boolean printVersionData) | Enable/disable whether the program prints data about the software versions used to the console. (Enabled by default). |

**init()**

Code placed in the init() method will be run before the game loop begins. This method is commonly used to initialize variables and set up the game.

**tick(double delta)**

Code placed in the tick(double delta) method will be run every tick. This method is commonly used to update the game.

**render()**

Code placed in the render() method will be run every tick. This method is commonly used to perform rendering tasks.

Unlike init() and tick(double delta), this method is optional.

**Input**

Calling keyManager.KEY\_W, keyManager.key\_A, etc. will return true if that key is currently being pressed and false if it is not currently being pressed.

**Camera**

The camera is automatically accessible in the game engine.

The camera by default is positioned at (0, 0, -2). The camera position can be changed with setCameraPos(Vector3f cameraPos).

The camera can also be translated by using the method translate(float translateX, float translate, float translate).

**Meshes**

Two types of meshes can be created. The first is a Color Mesh and it can be created by using the constructor Mesh(float[] vertices, short[] indices, float[] colors). The second is a Texture Mesh and it can be created by using the constructor Mesh(float[] vertices, short[] indices, float[] textureCoords, int[] textureIDs).

Any meshes created will automatically tick and render so there is no need to manually call their tick and render methods.

Meshes can be translated with the method translate(float translateX, float translateY, float translate). Meshes can be rotated with the method rotate(float rotateX, float rotateY, float rotateZ). Meshes can be scaled with the method scale(float scaleX, float scaleY, float scaleZ).

Meshes can be set at a position with the method setLocation(Vector3f location). Meshes can be set at a rotation with the method setRotation(Vector3f rotation). Meshes can be set at a scale with the method setScale(Vector3f scale).