**Passed in the menuState into the setState method instead of the starterStage from RC 1.**

**package** sonar.gamestates;

**import** java.awt.Graphics;

**import** sonar.Game;

**import** sonar.gamestates.states.Inventory;

**import** sonar.gamestates.states.MenuState;

**import** sonar.gamestates.states.PasswordState;

**import** sonar.gamestates.states.StarterStage;

**public** **class** GSM

{

//The GSM class that allows us to switch between different gamestates.

**private** Game game;

**private** GameState currentState, pastState;

**public** GSM(Game currentGame)

{

game = currentGame;

setState(StateHolder.***menuState***);

}

**public** **void** switchStates(GameState active, GameState passive)

{

**if**(pastState != **null**)

{

currentState = active;

pastState = passive;

}

}

//A gsm is in charge of loading and setting states

**private** **void** loadState(**int** state)

{

currentState = **null**;

**if**(state == StateHolder.***menuState***) currentState = **new** MenuState(**new** SingleStateBuilder(), **this**);

**if**(state == StateHolder.***passwordState***) currentState = **new** PasswordState(**new** SingleStateBuilder(), **this**);

**if**(state == StateHolder.***starterStage***) currentState = **new** StarterStage(**new** DualStateBuilder(), **this**);

**if**(currentState.getBuildState().stateType().equals("Dual")){pastState = **new** Inventory(**new** DualStateBuilder(), **this**);}

}

**public** **void** setState(**int** state)

{

loadState(state);

}

**public** **void** update()

{

currentState.update();

}

**public** **void** render(Graphics g)

{

currentState.render(g);

}

**public** Game getGame(){**return** game;}

**public** GameState getCurrentState(){**return** currentState;}

**public** GameState getPastState(){**return** pastState;}

}