

COS214 Project

Generated by Doxygen 1.9.2

| | |
|--------------------------------------|-----------|
| 1 COS214 Project | 1 |
| 1.1 COS214 Group Project | 1 |
| 1.2 Group Members | 1 |
| 2 Hierarchical Index | 3 |
| 2.1 Class Hierarchy | 3 |
| 3 Class Index | 5 |
| 3.1 Class List | 5 |
| 4 File Index | 7 |
| 4.1 File List | 7 |
| 5 Class Documentation | 11 |
| 5.1 Broadcasting Class Reference | 11 |
| 5.1.1 Member Function Documentation | 11 |
| 5.1.1.1 getType() | 11 |
| 5.1.1.2 handleChange() | 11 |
| 5.2 Builder Class Reference | 12 |
| 5.2.1 Member Function Documentation | 12 |
| 5.2.1.1 buildFalcon9() | 12 |
| 5.2.1.2 buildFalconHeavy() | 12 |
| 5.2.1.3 constructCapsule() | 13 |
| 5.2.1.4 createSimulation() | 13 |
| 5.2.1.5 getSpacecraft() | 13 |
| 5.3 CapsuleArriving Class Reference | 13 |
| 5.3.1 Member Function Documentation | 13 |
| 5.3.1.1 getState() | 14 |
| 5.3.1.2 handleChange() | 14 |
| 5.4 CapsuleDeparting Class Reference | 14 |
| 5.4.1 Member Function Documentation | 14 |
| 5.4.1.1 getState() | 14 |
| 5.4.1.2 handleChange() | 15 |
| 5.5 CapsuleDocked Class Reference | 15 |
| 5.5.1 Member Function Documentation | 15 |
| 5.5.1.1 getState() | 15 |
| 5.5.1.2 handleChange() | 15 |
| 5.6 CapsuleOffline Class Reference | 16 |
| 5.6.1 Member Function Documentation | 16 |
| 5.6.1.1 getState() | 16 |
| 5.6.1.2 handleChange() | 16 |
| 5.7 CapsuleState Class Reference | 16 |
| 5.7.1 Member Function Documentation | 17 |
| 5.7.1.1 getState() | 17 |

| | |
|---|----|
| 5.7.1.2 handleChange() | 17 |
| 5.8 Caretaker Class Reference | 17 |
| 5.8.1 Member Function Documentation | 18 |
| 5.8.1.1 getSize() | 18 |
| 5.8.1.2 retrieveMemento() | 18 |
| 5.8.1.3 storeMemento() | 18 |
| 5.9 CargoDragon Class Reference | 18 |
| 5.9.1 Constructor & Destructor Documentation | 19 |
| 5.9.1.1 CargoDragon() | 19 |
| 5.9.2 Member Function Documentation | 19 |
| 5.9.2.1 simulate() | 19 |
| 5.9.2.2 test() | 19 |
| 5.10 CommNetwork Class Reference | 20 |
| 5.10.1 Member Function Documentation | 20 |
| 5.10.1.1 notify() | 20 |
| 5.10.1.2 sendMessage() | 20 |
| 5.10.2 Member Data Documentation | 21 |
| 5.10.2.1 colleagueList | 21 |
| 5.11 Component Class Reference | 21 |
| 5.11.1 Constructor & Destructor Documentation | 22 |
| 5.11.1.1 Component() | 22 |
| 5.11.2 Member Function Documentation | 22 |
| 5.11.2.1 add() | 22 |
| 5.11.2.2 fireMerlin() | 22 |
| 5.11.2.3 fireVacuumMerlin() | 22 |
| 5.11.2.4 getComponent() | 23 |
| 5.11.2.5 getCost() | 23 |
| 5.11.2.6 getSize() | 23 |
| 5.11.2.7 land() | 23 |
| 5.11.2.8 remove() | 23 |
| 5.11.2.9 separate() | 24 |
| 5.11.2.10 simulate() | 24 |
| 5.11.2.11 test() | 24 |
| 5.11.3 Member Data Documentation | 24 |
| 5.11.3.1 cost | 24 |
| 5.12 ComponentComposite Class Reference | 25 |
| 5.12.1 Member Function Documentation | 25 |
| 5.12.1.1 add() | 25 |
| 5.12.1.2 fireMerlin() | 26 |
| 5.12.1.3 getComponent() | 26 |
| 5.12.1.4 getSize() | 26 |
| 5.12.1.5 land() | 26 |

| | |
|---|----|
| 5.12.1.6 remove() | 26 |
| 5.12.1.7 separate() | 27 |
| 5.12.1.8 simulate() | 27 |
| 5.12.1.9 test() | 27 |
| 5.13 ComponentCreator Class Reference | 27 |
| 5.13.1 Member Function Documentation | 28 |
| 5.13.1.1 factoryMethod() | 28 |
| 5.14 ConcreteRocketBuilder Class Reference | 28 |
| 5.14.1 Member Function Documentation | 28 |
| 5.14.1.1 buildFalcon9() | 28 |
| 5.14.1.2 buildFalconHeavy() | 29 |
| 5.14.1.3 constructCapsule() | 29 |
| 5.14.1.4 createSimulation() | 29 |
| 5.14.1.5 getCapsule() | 29 |
| 5.14.1.6 getSpacecraft() | 29 |
| 5.15 CoreCreator Class Reference | 30 |
| 5.15.1 Member Function Documentation | 30 |
| 5.15.1.1 factoryMethod() | 30 |
| 5.16 CrewDragon Class Reference | 30 |
| 5.16.1 Constructor & Destructor Documentation | 31 |
| 5.16.1.1 CrewDragon() | 31 |
| 5.16.2 Member Function Documentation | 31 |
| 5.16.2.1 getPassengers() | 31 |
| 5.16.2.2 setPassengers() | 31 |
| 5.16.2.3 simulate() | 31 |
| 5.16.2.4 test() | 32 |
| 5.17 Director Class Reference | 32 |
| 5.17.1 Constructor & Destructor Documentation | 32 |
| 5.17.1.1 Director() | 32 |
| 5.17.2 Member Function Documentation | 33 |
| 5.17.2.1 construct() | 33 |
| 5.17.2.2 constructCapsule() | 33 |
| 5.17.2.3 createSimulation() | 33 |
| 5.18 Facade Class Reference | 33 |
| 5.18.1 Member Function Documentation | 34 |
| 5.18.1.1 build() | 34 |
| 5.18.1.2 deliverCrew() | 34 |
| 5.18.1.3 distributeSatellites() | 35 |
| 5.18.1.4 editSimulation() | 35 |
| 5.18.1.5 fireMerlin() | 35 |
| 5.18.1.6 getRocket() | 35 |
| 5.18.1.7 jettisonFairing() | 35 |

| | |
|---|----|
| 5.18.1.8 launch() | 35 |
| 5.18.1.9 printSimulation() | 36 |
| 5.18.1.10 retrieveAll() | 36 |
| 5.18.1.11 retrieveSimulation() | 36 |
| 5.18.1.12 runSimulation() | 36 |
| 5.18.1.13 separateBoosters() | 36 |
| 5.18.1.14 staticFireTest() | 36 |
| 5.18.1.15 storeSimulation() | 37 |
| 5.18.1.16 test() | 37 |
| 5.18.1.17 useCommNetwork() | 37 |
| 5.19 Fairing Class Reference | 37 |
| 5.19.1 Constructor & Destructor Documentation | 38 |
| 5.19.1.1 Fairing() | 38 |
| 5.19.2 Member Function Documentation | 38 |
| 5.19.2.1 getSatellite() | 38 |
| 5.19.2.2 getSatellites() | 38 |
| 5.19.2.3 setSatellites() | 38 |
| 5.19.2.4 simulate() | 39 |
| 5.19.2.5 test() | 39 |
| 5.20 FalconCore Class Reference | 39 |
| 5.20.1 Member Function Documentation | 40 |
| 5.20.1.1 land() | 40 |
| 5.20.1.2 separate() | 40 |
| 5.20.1.3 simulate() | 40 |
| 5.20.1.4 test() | 40 |
| 5.21 Mediator Class Reference | 41 |
| 5.21.1 Member Function Documentation | 41 |
| 5.21.1.1 notify() | 41 |
| 5.21.1.2 sendMessage() | 41 |
| 5.22 Memento Class Reference | 43 |
| 5.22.1 Member Function Documentation | 43 |
| 5.22.1.1 getState() | 43 |
| 5.22.1.2 setState() | 43 |
| 5.23 MerlinEngine Class Reference | 44 |
| 5.23.1 Member Function Documentation | 44 |
| 5.23.1.1 fireMerlin() | 44 |
| 5.23.1.2 simulate() | 44 |
| 5.23.1.3 test() | 44 |
| 5.24 MerlinEngineCreator Class Reference | 45 |
| 5.24.1 Member Function Documentation | 45 |
| 5.24.1.1 factoryMethod() | 45 |
| 5.25 Observer Class Reference | 45 |

| | |
|---|----|
| 5.25.1 Member Function Documentation | 46 |
| 5.25.1.1 update() | 46 |
| 5.26 Offline Class Reference | 46 |
| 5.26.1 Member Function Documentation | 46 |
| 5.26.1.1 getType() | 46 |
| 5.26.1.2 handleChange() | 47 |
| 5.27 Online Class Reference | 47 |
| 5.27.1 Member Function Documentation | 47 |
| 5.27.1.1 getType() | 47 |
| 5.27.1.2 handleChange() | 47 |
| 5.28 RocketCapsule Class Reference | 48 |
| 5.28.1 Member Function Documentation | 48 |
| 5.28.1.1 addCapsule() | 48 |
| 5.28.1.2 getPayloadWeight() | 49 |
| 5.28.1.3 getSatellite() | 49 |
| 5.28.1.4 getState() | 49 |
| 5.28.1.5 requestStateChange() | 49 |
| 5.28.1.6 setPassengers() | 49 |
| 5.28.1.7 setPayloadWeight() | 50 |
| 5.28.1.8 setSatellites() | 50 |
| 5.28.1.9 setState() | 50 |
| 5.28.1.10 simulate() | 51 |
| 5.28.1.11 test() | 51 |
| 5.28.2 Member Data Documentation | 51 |
| 5.28.2.1 capsuleType | 51 |
| 5.28.2.2 state | 51 |
| 5.29 Satellite Class Reference | 51 |
| 5.29.1 Constructor & Destructor Documentation | 52 |
| 5.29.1.1 Satellite() | 52 |
| 5.29.2 Member Function Documentation | 53 |
| 5.29.2.1 attach() | 53 |
| 5.29.2.2 changed() | 53 |
| 5.29.2.3 clone() | 53 |
| 5.29.2.4 detach() | 53 |
| 5.29.2.5 getID() | 54 |
| 5.29.2.6 getState() | 54 |
| 5.29.2.7 notify() | 54 |
| 5.29.2.8 receiveMessage() | 54 |
| 5.29.2.9 requestStateChange() | 54 |
| 5.29.2.10 sendMessage() | 55 |
| 5.29.2.11 setMediator() | 55 |
| 5.29.2.12 setState() | 55 |

| | |
|---|----|
| 5.29.3 Member Data Documentation | 55 |
| 5.29.3.1 ID | 56 |
| 5.29.3.2 mediator | 56 |
| 5.29.3.3 observerList | 56 |
| 5.29.3.4 satelliteState | 56 |
| 5.30 SatelliteCreator Class Reference | 56 |
| 5.30.1 Member Function Documentation | 56 |
| 5.30.1.1 factoryMethod() | 56 |
| 5.30.1.2 setIDCount() | 57 |
| 5.31 SatelliteState Class Reference | 57 |
| 5.31.1 Member Function Documentation | 57 |
| 5.31.1.1 getType() | 57 |
| 5.31.1.2 handleChange() | 58 |
| 5.32 Simulation Class Reference | 58 |
| 5.32.1 Constructor & Destructor Documentation | 59 |
| 5.32.1.1 Simulation() | 59 |
| 5.32.2 Member Function Documentation | 59 |
| 5.32.2.1 addCall() | 59 |
| 5.32.2.2 changeSatelliteState() | 59 |
| 5.32.2.3 containsCall() | 60 |
| 5.32.2.4 createMemento() | 60 |
| 5.32.2.5 deliverCrew() | 60 |
| 5.32.2.6 distributeSatellites() | 60 |
| 5.32.2.7 fireMerlin() | 61 |
| 5.32.2.8 fireVacuumMerlin() | 61 |
| 5.32.2.9 getSimulationSize() | 61 |
| 5.32.2.10 getState() | 61 |
| 5.32.2.11 jettisonFairing() | 61 |
| 5.32.2.12 landBoosters() | 61 |
| 5.32.2.13 launch() | 62 |
| 5.32.2.14 printSimulation() | 62 |
| 5.32.2.15 removeStage() | 62 |
| 5.32.2.16 restoreMemento() | 62 |
| 5.32.2.17 runSimulation() | 63 |
| 5.32.2.18 sendMessage() | 63 |
| 5.32.2.19 separateBoosters() | 63 |
| 5.32.2.20 staticFireTest() | 63 |
| 5.32.2.21 swapStage() | 63 |
| 5.32.2.22 updateSimulationState() | 64 |
| 5.33 SimulationState Class Reference | 64 |
| 5.33.1 Member Function Documentation | 64 |
| 5.33.1.1 getCapsuleType() | 65 |

| | |
|---|-----------|
| 5.33.1.2 getMethodCalls() | 65 |
| 5.33.1.3 getPassengers() | 65 |
| 5.33.1.4 getPayloadWeight() | 65 |
| 5.33.1.5 getRocketType() | 65 |
| 5.33.1.6 getSatellites() | 65 |
| 5.33.1.7 setCapsuleType() | 65 |
| 5.33.1.8 setMethodCalls() | 66 |
| 5.33.1.9 setPassengers() | 66 |
| 5.33.1.10 setPayloadWeight() | 66 |
| 5.33.1.11 setRocketType() | 67 |
| 5.33.1.12 setSatellites() | 67 |
| 5.34 StarlinkCreator Class Reference | 67 |
| 5.34.1 Member Function Documentation | 68 |
| 5.34.1.1 factoryMethod() | 68 |
| 5.34.1.2 setIDCount() | 68 |
| 5.35 StarlinkSatellite Class Reference | 68 |
| 5.35.1 Constructor & Destructor Documentation | 69 |
| 5.35.1.1 StarlinkSatellite() [1/2] | 69 |
| 5.35.1.2 StarlinkSatellite() [2/2] | 69 |
| 5.35.2 Member Function Documentation | 69 |
| 5.35.2.1 clone() | 69 |
| 5.35.2.2 getState() | 69 |
| 5.35.2.3 setState() | 70 |
| 5.36 User Class Reference | 70 |
| 5.36.1 Constructor & Destructor Documentation | 70 |
| 5.36.1.1 User() | 70 |
| 5.36.2 Member Function Documentation | 71 |
| 5.36.2.1 update() | 71 |
| 5.37 VacuumMerlinEngine Class Reference | 71 |
| 5.37.1 Member Function Documentation | 71 |
| 5.37.1.1 fireVacuumMerlin() | 71 |
| 5.37.1.2 simulate() | 72 |
| 5.37.1.3 test() | 72 |
| 5.38 VacuumMerlinEngineCreator Class Reference | 72 |
| 5.38.1 Member Function Documentation | 72 |
| 5.38.1.1 factoryMethod() | 72 |
| 6 File Documentation | 73 |
| 6.1 C:/Users/labuser2/Downloads/System/System/Broadcasting.cpp File Reference | 73 |
| 6.1.1 Detailed Description | 73 |
| 6.2 C:/Users/labuser2/Downloads/System/System/Broadcasting.h File Reference | 73 |
| 6.2.1 Detailed Description | 73 |

| | |
|---|----|
| 6.3 Broadcasting.h | 73 |
| 6.4 C:/Users/labuser2/Downloads/System/System/Builder.h File Reference | 74 |
| 6.4.1 Detailed Description | 74 |
| 6.5 Builder.h | 74 |
| 6.6 C:/Users/labuser2/Downloads/System/System/CapsuleArriving.cpp File Reference | 74 |
| 6.6.1 Detailed Description | 75 |
| 6.7 C:/Users/labuser2/Downloads/System/System/CapsuleArriving.h File Reference | 75 |
| 6.7.1 Detailed Description | 75 |
| 6.8 CapsuleArriving.h | 75 |
| 6.9 C:/Users/labuser2/Downloads/System/System/CapsuleDeparting.cpp File Reference | 75 |
| 6.9.1 Detailed Description | 75 |
| 6.10 C:/Users/labuser2/Downloads/System/System/CapsuleDeparting.h File Reference | 76 |
| 6.10.1 Detailed Description | 76 |
| 6.11 CapsuleDeparting.h | 76 |
| 6.12 C:/Users/labuser2/Downloads/System/System/CapsuleDocked.cpp File Reference | 76 |
| 6.12.1 Detailed Description | 76 |
| 6.13 C:/Users/labuser2/Downloads/System/System/CapsuleDocked.h File Reference | 77 |
| 6.13.1 Detailed Description | 77 |
| 6.14 CapsuleDocked.h | 77 |
| 6.15 C:/Users/labuser2/Downloads/System/System/CapsuleOffline.cpp File Reference | 77 |
| 6.15.1 Detailed Description | 77 |
| 6.16 C:/Users/labuser2/Downloads/System/System/CapsuleOffline.h File Reference | 77 |
| 6.16.1 Detailed Description | 78 |
| 6.17 CapsuleOffline.h | 78 |
| 6.18 C:/Users/labuser2/Downloads/System/System/CapsuleState.h File Reference | 78 |
| 6.18.1 Detailed Description | 78 |
| 6.19 CapsuleState.h | 78 |
| 6.20 C:/Users/labuser2/Downloads/System/System/Caretaker.cpp File Reference | 79 |
| 6.20.1 Detailed Description | 79 |
| 6.21 C:/Users/labuser2/Downloads/System/System/Caretaker.h File Reference | 79 |
| 6.21.1 Detailed Description | 79 |
| 6.22 Caretaker.h | 79 |
| 6.23 C:/Users/labuser2/Downloads/System/System/CargoDragon.cpp File Reference | 80 |
| 6.23.1 Detailed Description | 80 |
| 6.24 C:/Users/labuser2/Downloads/System/System/CargoDragon.h File Reference | 80 |
| 6.24.1 Detailed Description | 80 |
| 6.25 CargoDragon.h | 80 |
| 6.26 C:/Users/labuser2/Downloads/System/System/CommNetwork.cpp File Reference | 81 |
| 6.26.1 Detailed Description | 81 |
| 6.27 C:/Users/labuser2/Downloads/System/System/CommNetwork.h File Reference | 81 |
| 6.27.1 Detailed Description | 81 |
| 6.28 CommNetwork.h | 81 |

| | |
|---|----|
| 6.29 C:/Users/labuser2/Downloads/System/System/Component.cpp File Reference | 82 |
| 6.29.1 Detailed Description | 82 |
| 6.30 C:/Users/labuser2/Downloads/System/System/Component.h File Reference | 82 |
| 6.30.1 Detailed Description | 82 |
| 6.31 Component.h | 82 |
| 6.32 C:/Users/labuser2/Downloads/System/System/ComponentComposite.cpp File Reference | 83 |
| 6.32.1 Detailed Description | 83 |
| 6.33 C:/Users/labuser2/Downloads/System/System/ComponentComposite.h File Reference | 83 |
| 6.33.1 Detailed Description | 83 |
| 6.34 ComponentComposite.h | 84 |
| 6.35 C:/Users/labuser2/Downloads/System/System/ComponentCreator.h File Reference | 84 |
| 6.35.1 Detailed Description | 84 |
| 6.36 ComponentCreator.h | 84 |
| 6.37 C:/Users/labuser2/Downloads/System/System/ConcreteRocketBuilder.cpp File Reference | 85 |
| 6.37.1 Detailed Description | 85 |
| 6.38 C:/Users/labuser2/Downloads/System/System/ConcreteRocketBuilder.h File Reference | 85 |
| 6.38.1 Detailed Description | 85 |
| 6.39 ConcreteRocketBuilder.h | 86 |
| 6.40 C:/Users/labuser2/Downloads/System/System/CoreCreator.cpp File Reference | 86 |
| 6.40.1 Detailed Description | 86 |
| 6.41 C:/Users/labuser2/Downloads/System/System/CoreCreator.h File Reference | 87 |
| 6.41.1 Detailed Description | 87 |
| 6.42 CoreCreator.h | 87 |
| 6.43 C:/Users/labuser2/Downloads/System/System/CrewDragon.cpp File Reference | 87 |
| 6.43.1 Detailed Description | 87 |
| 6.44 C:/Users/labuser2/Downloads/System/System/CrewDragon.h File Reference | 87 |
| 6.44.1 Detailed Description | 88 |
| 6.45 CrewDragon.h | 88 |
| 6.46 C:/Users/labuser2/Downloads/System/System/Director.cpp File Reference | 88 |
| 6.46.1 Detailed Description | 88 |
| 6.47 C:/Users/labuser2/Downloads/System/System/Director.h File Reference | 88 |
| 6.47.1 Detailed Description | 89 |
| 6.48 Director.h | 89 |
| 6.49 C:/Users/labuser2/Downloads/System/System/Facade.cpp File Reference | 89 |
| 6.49.1 Detailed Description | 89 |
| 6.50 C:/Users/labuser2/Downloads/System/System/Facade.h File Reference | 89 |
| 6.50.1 Detailed Description | 90 |
| 6.51 Facade.h | 90 |
| 6.52 C:/Users/labuser2/Downloads/System/System/Fairing.cpp File Reference | 91 |
| 6.52.1 Detailed Description | 91 |
| 6.53 C:/Users/labuser2/Downloads/System/System/Fairing.h File Reference | 91 |
| 6.53.1 Detailed Description | 91 |

| | | |
|--------|--|----|
| 6.54 | Fairing.h | 91 |
| 6.55 | C:/Users/labuser2/Downloads/System/System/FalconCore.cpp File Reference | 92 |
| 6.55.1 | Detailed Description | 92 |
| 6.56 | C:/Users/labuser2/Downloads/System/System/FalconCore.h File Reference | 92 |
| 6.56.1 | Detailed Description | 92 |
| 6.57 | FalconCore.h | 92 |
| 6.58 | C:/Users/labuser2/Downloads/System/System/main.cpp File Reference | 93 |
| 6.58.1 | Detailed Description | 93 |
| 6.59 | mainpage.h | 93 |
| 6.60 | C:/Users/labuser2/Downloads/System/System/Mediator.h File Reference | 93 |
| 6.60.1 | Detailed Description | 93 |
| 6.61 | Mediator.h | 93 |
| 6.62 | C:/Users/labuser2/Downloads/System/System/Memento.cpp File Reference | 94 |
| 6.62.1 | Detailed Description | 94 |
| 6.63 | C:/Users/labuser2/Downloads/System/System/Memento.h File Reference | 94 |
| 6.63.1 | Detailed Description | 94 |
| 6.64 | Memento.h | 94 |
| 6.65 | C:/Users/labuser2/Downloads/System/System/MerlinEngine.cpp File Reference | 95 |
| 6.65.1 | Detailed Description | 95 |
| 6.66 | C:/Users/labuser2/Downloads/System/System/MerlinEngine.h File Reference | 95 |
| 6.66.1 | Detailed Description | 95 |
| 6.67 | MerlinEngine.h | 95 |
| 6.68 | C:/Users/labuser2/Downloads/System/System/MerlinEngineCreator.cpp File Reference | 95 |
| 6.68.1 | Detailed Description | 96 |
| 6.69 | C:/Users/labuser2/Downloads/System/System/MerlinEngineCreator.h File Reference | 96 |
| 6.69.1 | Detailed Description | 96 |
| 6.70 | MerlinEngineCreator.h | 96 |
| 6.71 | C:/Users/labuser2/Downloads/System/System/Observer.h File Reference | 96 |
| 6.71.1 | Detailed Description | 97 |
| 6.72 | Observer.h | 97 |
| 6.73 | C:/Users/labuser2/Downloads/System/System/Offline.cpp File Reference | 97 |
| 6.73.1 | Detailed Description | 97 |
| 6.74 | C:/Users/labuser2/Downloads/System/System/Offline.h File Reference | 97 |
| 6.74.1 | Detailed Description | 97 |
| 6.75 | Offline.h | 98 |
| 6.76 | C:/Users/labuser2/Downloads/System/System/Online.cpp File Reference | 98 |
| 6.76.1 | Detailed Description | 98 |
| 6.77 | C:/Users/labuser2/Downloads/System/System/Online.h File Reference | 98 |
| 6.77.1 | Detailed Description | 98 |
| 6.78 | Online.h | 98 |
| 6.79 | C:/Users/labuser2/Downloads/System/System/RocketCapsule.cpp File Reference | 99 |
| 6.79.1 | Detailed Description | 99 |

| | |
|---|-----|
| 6.80 C:/Users/labuser2/Downloads/System/System/RocketCapsule.h File Reference | 99 |
| 6.80.1 Detailed Description | 99 |
| 6.81 RocketCapsule.h | 99 |
| 6.82 C:/Users/labuser2/Downloads/System/System/Satellite.cpp File Reference | 100 |
| 6.82.1 Detailed Description | 100 |
| 6.83 C:/Users/labuser2/Downloads/System/System/Satellite.h File Reference | 100 |
| 6.83.1 Detailed Description | 101 |
| 6.84 Satellite.h | 101 |
| 6.85 C:/Users/labuser2/Downloads/System/System/SatelliteCreator.cpp File Reference | 101 |
| 6.85.1 Detailed Description | 102 |
| 6.86 C:/Users/labuser2/Downloads/System/System/SatelliteCreator.h File Reference | 102 |
| 6.86.1 Detailed Description | 102 |
| 6.87 SatelliteCreator.h | 102 |
| 6.88 C:/Users/labuser2/Downloads/System/System/SatelliteState.h File Reference | 102 |
| 6.88.1 Detailed Description | 103 |
| 6.89 SatelliteState.h | 103 |
| 6.90 C:/Users/labuser2/Downloads/System/System/Simulation.cpp File Reference | 103 |
| 6.90.1 Detailed Description | 103 |
| 6.90.2 Variable Documentation | 104 |
| 6.90.2.1 rocket | 104 |
| 6.91 C:/Users/labuser2/Downloads/System/System/Simulation.h File Reference | 104 |
| 6.91.1 Detailed Description | 104 |
| 6.92 Simulation.h | 104 |
| 6.93 C:/Users/labuser2/Downloads/System/System/SimulationState.cpp File Reference | 105 |
| 6.93.1 Detailed Description | 105 |
| 6.94 C:/Users/labuser2/Downloads/System/System/SimulationState.h File Reference | 106 |
| 6.94.1 Detailed Description | 106 |
| 6.95 SimulationState.h | 106 |
| 6.96 C:/Users/labuser2/Downloads/System/System/StarlinkCreator.cpp File Reference | 107 |
| 6.96.1 Detailed Description | 107 |
| 6.97 C:/Users/labuser2/Downloads/System/System/StarlinkCreator.h File Reference | 107 |
| 6.97.1 Detailed Description | 107 |
| 6.98 StarlinkCreator.h | 107 |
| 6.99 C:/Users/labuser2/Downloads/System/System/StarlinkSatellite.cpp File Reference | 108 |
| 6.99.1 Detailed Description | 108 |
| 6.100 C:/Users/labuser2/Downloads/System/System/StarlinkSatellite.h File Reference | 108 |
| 6.100.1 Detailed Description | 108 |
| 6.101 StarlinkSatellite.h | 108 |
| 6.102 C:/Users/labuser2/Downloads/System/System/User.cpp File Reference | 108 |
| 6.102.1 Detailed Description | 109 |
| 6.103 C:/Users/labuser2/Downloads/System/System/User.h File Reference | 109 |
| 6.103.1 Detailed Description | 109 |

| | |
|--|------------|
| 6.104 User.h | 109 |
| 6.105 C:/Users/labuser2/Downloads/System/System/VacuumMerlinEngine.cpp File Reference | 109 |
| 6.105.1 Detailed Description | 110 |
| 6.106 C:/Users/labuser2/Downloads/System/System/VacuumMerlinEngine.h File Reference | 110 |
| 6.106.1 Detailed Description | 110 |
| 6.107 VacuumMerlinEngine.h | 110 |
| 6.108 C:/Users/labuser2/Downloads/System/System/VacuumMerlinEngineCreator.h File Reference | 110 |
| 6.108.1 Detailed Description | 111 |
| 6.109 VacuumMerlinEngineCreator.h | 111 |
| Index | 113 |

Chapter 1

COS214 Project

Authors

The 6 Musketeers

1.1 COS214 Group Project

This is the documentation for the COS214 Project 2021.

1.2 Group Members

u20632429 - Chiara Goncalves u20444738 - Zoe Liebenberg u20438151 - Jade Peché u17030553 - Ben Pietersen
u20498510 - Dylan Pietersen u20430516 - Steven Schormann

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

| | |
|-------------------------------------|----|
| Builder | 12 |
| ConcreteRocketBuilder | 28 |
| CapsuleState | 16 |
| CapsuleArriving | 13 |
| CapsuleDeparting | 14 |
| CapsuleDocked | 15 |
| CapsuleOffline | 16 |
| Caretaker | 17 |
| Component | 21 |
| ComponentComposite | 25 |
| FalconCore | 39 |
| MerlinEngine | 44 |
| RocketCapsule | 48 |
| CargoDragon | 18 |
| CrewDragon | 30 |
| Fairing | 37 |
| VacuumMerlinEngine | 71 |
| ComponentCreator | 27 |
| CoreCreator | 30 |
| MerlinEngineCreator | 45 |
| VacuumMerlinEngineCreator | 72 |
| Director | 32 |
| Facade | 33 |
| Mediator | 41 |
| CommNetwork | 20 |
| Memento | 43 |
| Observer | 45 |
| User | 70 |
| Satellite | 51 |
| StarlinkSatellite | 68 |
| SatelliteCreator | 56 |
| StarlinkCreator | 67 |
| SatelliteState | 57 |
| Broadcasting | 11 |
| Offline | 46 |
| Online | 47 |
| Simulation | 58 |
| SimulationState | 64 |

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

| | |
|---------------------------|----|
| Broadcasting | 11 |
| Builder | 12 |
| CapsuleArriving | 13 |
| CapsuleDeparting | 14 |
| CapsuleDocked | 15 |
| CapsuleOffline | 16 |
| CapsuleState | 16 |
| Caretaker | 17 |
| CargoDragon | 18 |
| CommNetwork | 20 |
| Component | 21 |
| ComponentComposite | 25 |
| ComponentCreator | 27 |
| ConcreteRocketBuilder | 28 |
| CoreCreator | 30 |
| CrewDragon | 30 |
| Director | 32 |
| Facade | 33 |
| Fairing | 37 |
| FalconCore | 39 |
| Mediator | 41 |
| Memento | 43 |
| MerlinEngine | 44 |
| MerlinEngineCreator | 45 |
| Observer | 45 |
| Offline | 46 |
| Online | 47 |
| RocketCapsule | 48 |
| Satellite | 51 |
| SatelliteCreator | 56 |
| SatelliteState | 57 |
| Simulation | 58 |
| SimulationState | 64 |
| StarlinkCreator | 67 |
| StarlinkSatellite | 68 |
| User | 70 |
| VacuumMerlinEngine | 71 |
| VacuumMerlinEngineCreator | 72 |

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

| | |
|---|----|
| C:/Users/labuser2/Downloads/System/System/Broadcasting.cpp | |
| Implementation for Broadcasting.h | 73 |
| C:/Users/labuser2/Downloads/System/System/Broadcasting.h | |
| Participant - Concrete State (State). Describes the properties and methods of a Satellite in the 'broadcasting' state | 73 |
| C:/Users/labuser2/Downloads/System/System/Builder.h | |
| Participant - Builder (Builder) Describes the methods to build the components of a rocket | 74 |
| C:/Users/labuser2/Downloads/System/System/CapsuleArriving.cpp | |
| Implementation for CapsuleArriving.h | 74 |
| C:/Users/labuser2/Downloads/System/System/CapsuleArriving.h | |
| Participant - Concrete State (State) Describes the methods of a Capsule that is in an 'arriving' state | 75 |
| C:/Users/labuser2/Downloads/System/System/CapsuleDeparting.cpp | |
| Implementation for CapsuleDeparting.h | 75 |
| C:/Users/labuser2/Downloads/System/System/CapsuleDeparting.h | |
| Participant - Concrete State (State) Describes the methods of a Capsule that is in a 'departing' state | 76 |
| C:/Users/labuser2/Downloads/System/System/CapsuleDocked.cpp | |
| Implementation for CapsuleDocked.h | 76 |
| C:/Users/labuser2/Downloads/System/System/CapsuleDocked.h | |
| Participant - Concrete State (State) Describes the methods of a Capsule that is in a 'docked' state | 77 |
| C:/Users/labuser2/Downloads/System/System/CapsuleOffline.cpp | |
| Implementation for CapsuleOffline.h | 77 |
| C:/Users/labuser2/Downloads/System/System/CapsuleOffline.h | |
| Participant - Concrete State (State) Describes the methods of a Capsule that is in a 'docked' state | 77 |
| C:/Users/labuser2/Downloads/System/System/CapsuleState.h | |
| Participant - State (State) Describes the interface for the different states of a capsule | 78 |
| C:/Users/labuser2/Downloads/System/System/Caretaker.cpp | |
| Implementation for Caretaker.h | 79 |
| C:/Users/labuser2/Downloads/System/System/Caretaker.h | |
| Participant - Caretaker (Memento) Describes the class responsible for the safekeeping of the Memento class' state | 79 |
| C:/Users/labuser2/Downloads/System/System/CargoDragon.cpp | |
| Implementation for CargoDragon.h | 80 |

| | |
|--|----|
| C:/Users/labuser2/Downloads/System/System/CargoDragon.h | |
| Participant - Concrete Decorator (Decorator) Defines the attributes and methods for a RocketCapsule that carries Cargo | 80 |
| C:/Users/labuser2/Downloads/System/System/CommNetwork.cpp | |
| Implementation for CommNetwork.h | 81 |
| C:/Users/labuser2/Downloads/System/System/CommNetwork.h | |
| Participant - Concrete Mediator (Mediator) Defines the attributes and methods for the class used for communication between the satellites | 81 |
| C:/Users/labuser2/Downloads/System/System/Component.cpp | |
| Implementation for Component.h | 82 |
| C:/Users/labuser2/Downloads/System/System/Component.h | |
| Participant - Component (Decorator), Component (Composite), Client (Chain of Responsibility), Product (Builder), Product (Factory Method), Client (Prototype), Implementor (Bridge) Defines the attributes and methods for Component objects | 82 |
| C:/Users/labuser2/Downloads/System/System/ComponentComposite.cpp | |
| Implementation for ComponentComposite.h | 83 |
| C:/Users/labuser2/Downloads/System/System/ComponentComposite.h | |
| Participant - Handler (Chain of Responsibility) Defines an interface to handle the requests and implementatins of the Falcon9 and FalconHeavy components | 83 |
| C:/Users/labuser2/Downloads/System/System/ComponentCreator.h | |
| Participant - Creator (Factory Method), Prototype (Prototype). Defines the interface creating the Merlin, Vacuum Merlin and Core Engines | 84 |
| C:/Users/labuser2/Downloads/System/System/ConcreteRocketBuilder.cpp | |
| Implementation for ConcreteRocketBuilder.h | 85 |
| C:/Users/labuser2/Downloads/System/System/ConcreteRocketBuilder.h | |
| Participant - Concrete Builder (Builder) Defines the methods and attributes of the class that builds a rocket | 85 |
| C:/Users/labuser2/Downloads/System/System/CoreCreator.cpp | |
| Implementation for Cor.Creatorh | 86 |
| C:/Users/labuser2/Downloads/System/System/CoreCreator.h | |
| Participant - Concrete Creator (Factory Method), ConcretePrototype (Prototype). Defines the methods and attributes of the class that builds FalconCore engines | 87 |
| C:/Users/labuser2/Downloads/System/System/CrewDragon.cpp | |
| Implementation for CrewDragon.h | 87 |
| C:/Users/labuser2/Downloads/System/System/CrewDragon.h | |
| Participant - Concrete Decorator (Decorator) Defines the attributes and methods for a Capsule carrying crew members | 87 |
| C:/Users/labuser2/Downloads/System/System/Director.cpp | |
| Implementation for Director.h | 88 |
| C:/Users/labuser2/Downloads/System/System/Director.h | |
| Participant - Director (Builder) Defines the attributes and methods for the class that constructs rockets using the Builder interface | 88 |
| C:/Users/labuser2/Downloads/System/System/Facade.cpp | |
| Implementation for Facade.h | 89 |
| C:/Users/labuser2/Downloads/System/System/Facade.h | |
| Participant - Facade (Facade) Delegates client requests to appropriate subsystem objects . . . | 89 |
| C:/Users/labuser2/Downloads/System/System/Fairing.cpp | |
| Implementation for Fairing.h | 91 |
| C:/Users/labuser2/Downloads/System/System/Fairing.h | |
| Participant - Concrete Decorator (Decorator) Defines the attributes and methods for a RocketCapsule of type Fairing | 91 |
| C:/Users/labuser2/Downloads/System/System/FalconCore.cpp | |
| Implementation for FalconCore.h | 92 |
| C:/Users/labuser2/Downloads/System/System/FalconCore.h | |
| Participant - ConcreteComponent (Decorator), Leaf (Composite), ConcreteProduct (Factory Method), Concrete Implementor (Bridge) Defines the methods of the class that defines a FalconCore engine | 92 |

| | |
|---|-----|
| C:/Users/labuser2/Downloads/System/System/main.cpp | |
| Runs the program | 93 |
| C:/Users/labuser2/Downloads/System/System/mainpage.h | 93 |
| C:/Users/labuser2/Downloads/System/System/Mediator.h | |
| Participant - Mediator (Mediator) Defines the methods of the interface that enables communication between the different satellites | 93 |
| C:/Users/labuser2/Downloads/System/System/Memento.cpp | |
| Implementation for Memento.h | 94 |
| C:/Users/labuser2/Downloads/System/System/Memento.h | |
| Participant - Memento (Memento) Defines the methods of the class that stores the state of the simulation of a rocket | 94 |
| C:/Users/labuser2/Downloads/System/System/MerlinEngine.cpp | |
| Implementation for MerlinEngine.h | 95 |
| C:/Users/labuser2/Downloads/System/System/MerlinEngine.h | |
| Participant - ConcreteProduct (Factory Method), Concrete Implementor (Bridge). Defines the methods of the class that defines a Merlin engine | 95 |
| C:/Users/labuser2/Downloads/System/System/MerlinEngineCreator.cpp | |
| Implementation for MerlinEngineCreator.h | 95 |
| C:/Users/labuser2/Downloads/System/System/MerlinEngineCreator.h | |
| Participant - ConcreteCreator (Factory Method), ConcretePrototype (Prototype) Defines the methods of the class that creates MerlinEngine objects | 96 |
| C:/Users/labuser2/Downloads/System/System/Observer.h | |
| Participant - Observer (Observer) Defines the methods of the abstract class that observes the state of a satellite | 96 |
| C:/Users/labuser2/Downloads/System/System/Offline.cpp | |
| Implementation for Offline.h | 97 |
| C:/Users/labuser2/Downloads/System/System/Offline.h | |
| Participant - Concrete State (State). Describes the properties and methods of a Satellite in the 'Offline' state | 97 |
| C:/Users/labuser2/Downloads/System/System/Online.cpp | |
| Implementation for Online.h | 98 |
| C:/Users/labuser2/Downloads/System/System/Online.h | |
| Participant - Concrete State (State). Describes the properties and methods of a Satellite in the 'Online' state | 98 |
| C:/Users/labuser2/Downloads/System/System/RocketCapsule.cpp | |
| Implementation for RocketCapsule.h | 99 |
| C:/Users/labuser2/Downloads/System/System/RocketCapsule.h | |
| Participant - Decorator (Decorator), Context (State) Describes the properties and methods of a RocketCapule that can be added to a rocket Component | 99 |
| C:/Users/labuser2/Downloads/System/System/Satellite.cpp | |
| Implementation for Satellite.h | 100 |
| C:/Users/labuser2/Downloads/System/System/Satellite.h | |
| Participant - ConcreteSubject (Observer), Colleague (Mediator), Context (State), Product (Factory Method). Describes the properties and methods of a Satellite object | 100 |
| C:/Users/labuser2/Downloads/System/System/SatelliteCreator.cpp | |
| Implementation for SatelliteCreator.h | 101 |
| C:/Users/labuser2/Downloads/System/System/SatelliteCreator.h | |
| Participant - Creator (Factory Method). Describes the properties and methods the abstract class that creates a Satellite | 102 |
| C:/Users/labuser2/Downloads/System/System/SatelliteState.h | |
| Participant - State (State) Describes the interface for the different states of a Satellite | 102 |
| C:/Users/labuser2/Downloads/System/System/Simulation.cpp | |
| Implementation for Simulation.h | 103 |
| C:/Users/labuser2/Downloads/System/System/Simulation.h | |
| Participant - Abstraction (Bridge) Describes the abstract class for running a simulation of a rocket launch | 104 |

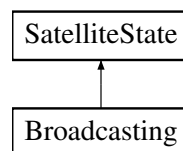
| | |
|--|-----|
| C:/Users/labuser2/Downloads/System/System/ SimulationState.cpp | |
| Implementation for SimulationState.h | 105 |
| C:/Users/labuser2/Downloads/System/System/ SimulationState.h | |
| Participant - State (Memento) Describes the attributes and methods of a SimulationState object | 106 |
| C:/Users/labuser2/Downloads/System/System/ StarlinkCreator.cpp | |
| Implementation for StarlinkCreator.h | 107 |
| C:/Users/labuser2/Downloads/System/System/ StarlinkCreator.h | |
| Participant - Concrete Creator (Factory Method) Describes the attributes and methods of the class that creates StarlinkSatellite objects | 107 |
| C:/Users/labuser2/Downloads/System/System/ StarlinkSatellite.cpp | |
| Implementation for StarlinkSatellite.h | 108 |
| C:/Users/labuser2/Downloads/System/System/ StarlinkSatellite.h | |
| Participant - Concrete Product (Factory Method) Describes the attributes and methods of StarlinkSatellite objects | 108 |
| C:/Users/labuser2/Downloads/System/System/ User.cpp | |
| Implementation for User.h | 108 |
| C:/Users/labuser2/Downloads/System/System/ User.h | |
| Participant - Concrete Observer (Observer) Describes the attributes and methods of class that observes the state of Satellite objects | 109 |
| C:/Users/labuser2/Downloads/System/System/ VacuumMerlinEngine.cpp | |
| Implementation for VacuumMerlinEngine.h | 109 |
| C:/Users/labuser2/Downloads/System/System/ VacuumMerlinEngine.h | |
| Participant - Concrete Product (Factory Method) Describes the attributes and methods of a VacuumMerlinEngine object | 110 |
| C:/Users/labuser2/Downloads/System/System/ VacuumMerlinEngineCreator.h | |
| Participant - ConcretePrototype (Prototpe), Concrete Implementor (Bridge), Concrete Product (Factory Method). Describes the attributes and methods of the class to create VacuumMerlinEngine objects | 110 |

Chapter 5

Class Documentation

5.1 Broadcasting Class Reference

Inheritance diagram for Broadcasting:



Public Member Functions

- **Broadcasting** ()
Constructor for the [Broadcasting](#) object.
- string [getType](#) ()
Returns the type of state the satellite is currently in ([Broadcasting](#)).
- [SatelliteState](#) * [handleChange](#) ()
Handles a change in state - sets the current state of the satellite to offline.

5.1.1 Member Function Documentation

5.1.1.1 `getType()`

```
string Broadcasting::getType ( ) [virtual]
```

Returns the type of state the satellite is currently in ([Broadcasting](#)).

Returns

string

Implements [SatelliteState](#).

5.1.1.2 `handleChange()`

```
SatelliteState * Broadcasting::handleChange ( ) [virtual]
```

Handles a change in state - sets the current state of the satellite to offline.

Returns

SatelliteState*

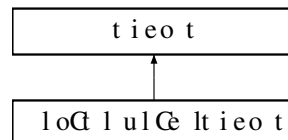
Implements [SatelliteState](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Broadcasting.h](#)
- C:/Users/labuser2/Downloads/System/System/[Broadcasting.cpp](#)

5.2 Builder Class Reference

Inheritance diagram for Builder:



Public Member Functions

- virtual void [buildFalcon9](#) ()=0
Pure virtual function to be implemented in children classes. The function builds a Falcon9 Core for the rocket.
- virtual void [buildFalconHeavy](#) ()=0
Pure virtual function to be implemented in children classes. The function builds a FalconHeavy Core for the rocket.
- virtual void [constructCapsule](#) (string c)=0
Pure virtual function to be implemented in children classes. The function constructs a capsule for the rocket.
- virtual [Component](#) * [getSpacecraft](#) ()=0
Pure virtual function to be implemented in children classes. The function returns the current spacecraft.
- virtual [Simulation](#) * [createSimulation](#) ()=0
Pure virtual function to be implemented in children classes. The function creates a simulation for the rocket.

5.2.1 Member Function Documentation

5.2.1.1 buildFalcon9()

```
virtual void Builder::buildFalcon9 ( ) [pure virtual]
```

Pure virtual function to be implemented in children classes. The function builds a Falcon9 Core for the rocket.

Returns

void

Implemented in [ConcreteRocketBuilder](#).

5.2.1.2 buildFalconHeavy()

```
virtual void Builder::buildFalconHeavy ( ) [pure virtual]
```

Pure virtual function to be implemented in children classes. The function builds a FalconHeavy Core for the rocket.

Returns

void

Implemented in [ConcreteRocketBuilder](#).

5.2.1.3 constructCapsule()

```
virtual void Builder::constructCapsule (
    string c ) [pure virtual]
```

Pure virtual function to be implemented in children classes. The function constructs a capsule for the rocket.

Returns

void

Implemented in [ConcreteRocketBuilder](#).

5.2.1.4 createSimulation()

```
virtual Simulation * Builder::createSimulation ( ) [pure virtual]
```

Pure virtual function to be implemented in children classes. The function creates a simulation for the rocket.

Returns

Simulation*

Implemented in [ConcreteRocketBuilder](#).

5.2.1.5 getSpacecraft()

```
virtual Component * Builder::getSpacecraft ( ) [pure virtual]
```

Pure virtual function to be implemented in children classes. The function returns the current spacecraft.

Returns

Component*

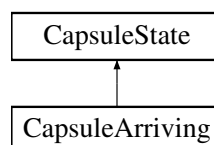
Implemented in [ConcreteRocketBuilder](#).

The documentation for this class was generated from the following file:

- C:/Users/labuser2/Downloads/System/System/[Builder.h](#)

5.3 CapsuleArriving Class Reference

Inheritance diagram for CapsuleArriving:



Public Member Functions

- **CapsuleArriving ()**
Constructor for [CapsuleArriving](#) objects.
- string **getState ()**
Returns the state that the capsule is currently in (Arriving).
- [CapsuleState](#) * **handleChange ()**
Handles a change in state - sets the state of the current capsule to 'docked'.

5.3.1 Member Function Documentation

5.3.1.1 getState()

```
string CapsuleArriving::getState ( ) [virtual]
```

Returns the state that the capsule is currently in (Arriving).

Returns

string

Implements [CapsuleState](#).

5.3.1.2 handleChange()

```
CapsuleState * CapsuleArriving::handleChange ( ) [virtual]
```

Handles a change in state - sets the state of the current capsule to 'docked'.

Returns

CapsuleState*

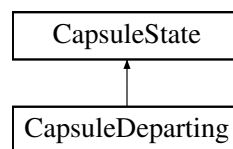
Implements [CapsuleState](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[CapsuleArriving.h](#)
- C:/Users/labuser2/Downloads/System/System/[CapsuleArriving.cpp](#)

5.4 CapsuleDeparting Class Reference

Inheritance diagram for CapsuleDeparting:



Public Member Functions

- **CapsuleDeparting ()**
Constructor for [CapsuleDeparting](#) objects.
- string [getState \(\)](#)
Returns the state that the capsule is currently in (Departing).
- [CapsuleState *](#) [handleChange \(\)](#)
Handles a change in state - sets the state of the current capsule to 'arriving'.

5.4.1 Member Function Documentation

5.4.1.1 getState()

```
string CapsuleDeparting::getState ( ) [virtual]
```

Returns the state that the capsule is currently in (Departing).

Returns

string

Implements [CapsuleState](#).

5.4.1.2 handleChange()

`CapsuleState * CapsuleDeparting::handleChange () [virtual]`

Handles a change in state - sets the state of the current capsule to 'arriving'.

Returns

`CapsuleState*`

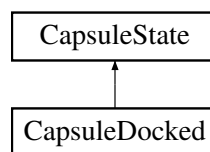
Implements [CapsuleState](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[CapsuleDeparting.h](#)
- C:/Users/labuser2/Downloads/System/System/[CapsuleDeparting.cpp](#)

5.5 CapsuleDocked Class Reference

Inheritance diagram for CapsuleDocked:



Public Member Functions

- **CapsuleDocked ()**
Constructor for [CapsuleDocked](#) objects.
- string **getState ()**
Returns the state that the capsule is currently in (Docked).
- `CapsuleState * handleChange \(\)`
Handles a change in state - sets the state of the current capsule to 'offline'.

5.5.1 Member Function Documentation

5.5.1.1 getState()

`string CapsuleDocked::getState () [virtual]`

Returns the state that the capsule is currently in (Docked).

Returns

`string`

Implements [CapsuleState](#).

5.5.1.2 handleChange()

`CapsuleState * CapsuleDocked::handleChange () [virtual]`

Handles a change in state - sets the state of the current capsule to 'offline'.

Returns

`CapsuleState*`

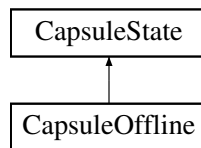
Implements [CapsuleState](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[CapsuleDocked.h](#)
- C:/Users/labuser2/Downloads/System/System/[CapsuleDocked.cpp](#)

5.6 CapsuleOffline Class Reference

Inheritance diagram for CapsuleOffline:



Public Member Functions

- **CapsuleOffline ()**
Constructor for [CapsuleOffline](#) objects.
- string **getState ()**
Returns the state that the capsule is currently in ([Offline](#)).
- [CapsuleState](#) * **handleChange ()**
Handles a change in state - sets the state of the current capsule to 'null'.

5.6.1 Member Function Documentation

5.6.1.1 getState()

```
string CapsuleOffline::getState ( ) [virtual]
```

Returns the state that the capsule is currently in ([Offline](#)).

Returns

string

Implements [CapsuleState](#).

5.6.1.2 handleChange()

```
CapsuleState * CapsuleOffline::handleChange ( ) [virtual]
```

Handles a change in state - sets the state of the current capsule to 'null'.

Returns

void

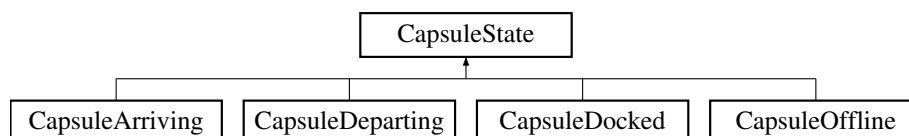
Implements [CapsuleState](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[CapsuleOffline.h](#)
- C:/Users/labuser2/Downloads/System/System/[CapsuleOffline.cpp](#)

5.7 CapsuleState Class Reference

Inheritance diagram for CapsuleState:



Public Member Functions

- virtual string [getState](#) ()=0
Pure virtual function to be implemented in children classes. The function returns the current state of the capsule (Arriving/Departing/Docked/Offline).
- virtual [CapsuleState](#) * [handleChange](#) ()=0
Pure virtual function to be implemented in children classes. Handles the change in state by setting the state of the capsule to a new state.

5.7.1 Member Function Documentation

5.7.1.1 [getState](#)()

```
virtual string CapsuleState::getState ( ) [pure virtual]
```

Pure virtual function to be implemented in children classes. The function returns the current state of the capsule (Arriving/Departing/Docked/Offline).

Returns

string

Implemented in [CapsuleArriving](#), [CapsuleDeparting](#), [CapsuleDocked](#), and [CapsuleOffline](#).

5.7.1.2 [handleChange](#)()

```
virtual CapsuleState * CapsuleState::handleChange ( ) [pure virtual]
```

Pure virtual function to be implemented in children classes. Handles the change in state by setting the state of the capsule to a new state.

Returns

[CapsuleState](#)*

Implemented in [CapsuleArriving](#), [CapsuleDeparting](#), [CapsuleDocked](#), and [CapsuleOffline](#).

The documentation for this class was generated from the following file:

- C:/Users/labuser2/Downloads/System/System/[CapsuleState.h](#)

5.8 Caretaker Class Reference

Public Member Functions

- **Caretaker** ()
Constructor for [Caretaker](#) objects. Sets the store to NULL.
- **~Caretaker** ()
Destructor for [Caretaker](#) objects. Deletes the store object and the memory allocated to it.
- void [storeMemento](#) ([Memento](#) *m)
Stores the current state of the rocket.
- [Memento](#) * [retrieveMemento](#) ()
Returns the current state of the rocket .
- int [getSize](#) ()
Returns the size of the store.

5.8.1 Member Function Documentation

5.8.1.1 getSize()

```
int Caretaker::getSize ( )
```

Returns the size of the store.

Returns

int

5.8.1.2 retrieveMemento()

```
Memento * Caretaker::retrieveMemento ( )
```

Returns the current state of the rocket .

Returns

Memento*

5.8.1.3 storeMemento()

```
void Caretaker::storeMemento (
    Memento * m )
```

Stores the current state of the rocket.

Parameters

| | |
|----------|---|
| <i>m</i> | Memento* - the Memento storing the current state of the rocket. |
|----------|---|

Returns

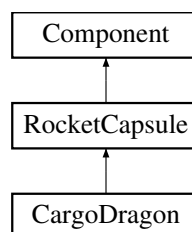
void

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Caretaker.h](#)
- C:/Users/labuser2/Downloads/System/System/[Caretaker.cpp](#)

5.9 CargoDragon Class Reference

Inheritance diagram for CargoDragon:



Public Member Functions

- [CargoDragon](#) ([Component](#) *r)

Constructor for [CargoDragon](#) objects. Takes in a rocket as a parameter and uses the [RocketCapsule\(parent\)](#) constructor to initialize the rocket variable.

- void [simulate](#) ()

Starts the simulation for [CargoDragon](#) objects.

- void [test](#) ()

Tests if the [CargoDragon](#) meets all the requirements for a successful launch. The requirements:

Additional Inherited Members

5.9.1 Constructor & Destructor Documentation

5.9.1.1 CargoDragon()

```
CargoDragon::CargoDragon (
    Component * r )
```

Constructor for [CargoDragon](#) objects. Takes in a rocket as a parameter and uses the [RocketCapsule\(parent\)](#) constructor to initialize the rocket variable.

Parameters

| | |
|----------|------------|
| <i>r</i> | Component* |
|----------|------------|

5.9.2 Member Function Documentation

5.9.2.1 simulate()

```
void CargoDragon::simulate ( ) [virtual]
```

Starts the simulation for [CargoDragon](#) objects.

Returns

void

Implements [RocketCapsule](#).

5.9.2.2 test()

```
void CargoDragon::test ( ) [virtual]
```

Tests if the [CargoDragon](#) meets all the requirements for a successful launch. The requirements:

- the cost must be >0
- it must have a capsuleType
- it must have a rocketType

Returns

void

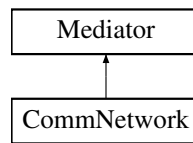
Implements [RocketCapsule](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[CargoDragon.h](#)
- C:/Users/labuser2/Downloads/System/System/[CargoDragon.cpp](#)

5.10 CommNetwork Class Reference

Inheritance diagram for CommNetwork:



Public Member Functions

- **CommNetwork** (vector< [Satellite](#) * >)
- void [notify](#) (int sender)
Notifies all the satellites (colleagues) if the state of the one of the satellites have changed.
- void [sendMessage](#) (int sender, int receiver, string msg)
Sends a string message to a particular satellite.

Public Attributes

- vector< [Satellite](#) * > [colleagueList](#)

5.10.1 Member Function Documentation

5.10.1.1 notify()

```
void CommNetwork::notify (
    int sender ) [virtual]
```

Notifies all the satellites (colleagues) if the state of the one of the satellites have changed.

Parameters

| | |
|------------------|--|
| <i>colleague</i> | Satellite* - the Satellite object that changed states. |
|------------------|--|

Returns

void

Implements [Mediator](#).

5.10.1.2 sendMessage()

```
void CommNetwork::sendMessage (
    int sender,
    int receiver,
    string msg ) [virtual]
```

Sends a string message to a particular satellite.

Parameters

| | |
|-----------------|-------------------------------|
| <i>sender</i> | int - The ID of the sender. |
| <i>receiver</i> | int - The ID of the receiver. |
| <i>msg</i> | string - The message to send. |

Returns

void

Implements [Mediator](#).

5.10.2 Member Data Documentation

5.10.2.1 colleagueList

```
vector<Satellite*> CommNetwork::colleagueList
```

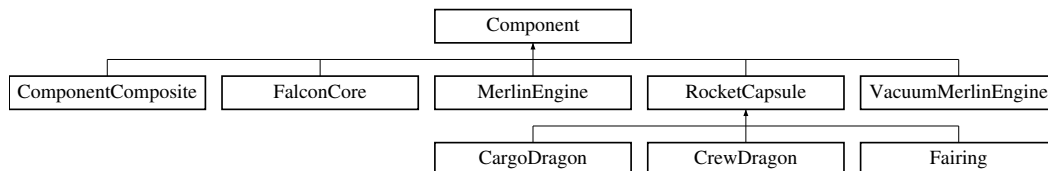
A vector of [Satellite](#) objects representing the colleagues of the [Mediator](#) pattern

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[CommNetwork.h](#)
- C:/Users/labuser2/Downloads/System/System/[CommNetwork.cpp](#)

5.11 Component Class Reference

Inheritance diagram for Component:



Public Member Functions

- [Component](#) (double c)
Constructor for [Component](#) objects. Takes in the cost as a parameter and initializes the cost variable.
- virtual void [simulate](#) ()
Virtual function that needs to be implemented in all the children classes. Starts the simulation for [Component](#) objects.
- virtual void [test](#) ()
Virtual function that tests if the [Component](#) meets all the requirements for a successful launch. The requirements depend on the type of [Component](#).
- virtual void [add](#) ([Component](#) *c)
Adds a component to the rocket.
- virtual void [remove](#) (int pos)
Virtual method that removes a component from the rocket based on its position.
- virtual [Component](#) * [getComponent](#) (int pos)
Virtual method that returns a component of the rocket based on its position.
- double [getCost](#) ()
Returns the cost of the component.
- virtual void [separate](#) ()
Virtual function that separates the [Component](#) from the rocket.
- virtual void [fireMerlin](#) ()
Virtual method to ignite a [MerlinEngine](#) object.
- virtual void [land](#) ()
Virtual method called when a [Component](#) object lands.
- virtual int [getSize](#) ()
Virtual method that returns the size of the [Component](#). return @int.
- virtual void [fireVacuumMerlin](#) ()
Virtual method to ignite a [VacuumMerlin](#) object.

Protected Attributes

- double [cost](#)

5.11.1 Constructor & Destructor Documentation

5.11.1.1 Component()

```
Component::Component (
    double c )
```

Constructor for [Component](#) objects. Takes in the cost as a parameter and initializes the cost variable.

Parameters

| | |
|----------------|--------------------------------------|
| <code>c</code> | double - the cost of the compoenent. |
|----------------|--------------------------------------|

5.11.2 Member Function Documentation

5.11.2.1 add()

```
void Component::add (
    Component * c ) [virtual]
```

Adds a component to the rocket.

Parameters

| | |
|----------------|--|
| <code>c</code> | Component* - the Component to add to the rocket. |
|----------------|--|

Returns

void

Reimplemented in [ComponentComposite](#).

5.11.2.2 fireMerlin()

```
void Component::fireMerlin ( ) [virtual]
```

Virtual method to ignite a [MerlinEngine](#) object.

Returns

void

Reimplemented in [ComponentComposite](#), and [MerlinEngine](#).

5.11.2.3 fireVacuumMerlin()

```
void Component::fireVacuumMerlin ( ) [virtual]
```

Virtual method to ignite a VacuumMerlin object.

Returns

void

Reimplemented in [VacuumMerlinEngine](#).

5.11.2.4 GetComponent()

```
Component * Component::GetComponent (
    int pos ) [virtual]
```

Virtual method that returns a component of the rocket based on its position.

Parameters

| | |
|------------|--|
| <i>pos</i> | int - the position of the Component in the vector of components. |
|------------|--|

Returns

Component*

Reimplemented in [ComponentComposite](#).

5.11.2.5 getCost()

```
double Component::getCost ( )
```

Returns the cost of the component.

Returns

double

5.11.2.6 getSize()

```
int Component::getSize ( ) [virtual]
```

Virtual method that returns the size of the [Component](#). return @int.

Reimplemented in [ComponentComposite](#).

5.11.2.7 land()

```
void Component::land ( ) [virtual]
```

Virtual method called when a [Component](#) object lands.

Returns

void

Reimplemented in [ComponentComposite](#), and [FalconCore](#).

5.11.2.8 remove()

```
void Component::remove (
    int pos ) [virtual]
```

Virtual method that removes a component from the rocket based on its position.

Parameters

| | |
|------------|--|
| <i>pos</i> | int - the position of the Component in the vector of components. |
|------------|--|

Returns

void

Reimplemented in [ComponentComposite](#).

5.11.2.9 separate()

```
void Component::separate ( ) [virtual]
```

Virtual function that separates the [Component](#) from the rocket.

Returns

void

Reimplemented in [ComponentComposite](#), and [FalconCore](#).

5.11.2.10 simulate()

```
void Component::simulate ( ) [virtual]
```

Virtual function that needs to be implemented in all the children classes. Starts the simulation for [Component](#) objects.

Returns

void

Reimplemented in [CargoDragon](#), [ComponentComposite](#), [CrewDragon](#), [Fairing](#), [FalconCore](#), [MerlinEngine](#), [VacuumMerlinEngine](#), and [RocketCapsule](#).

5.11.2.11 test()

```
void Component::test ( ) [virtual]
```

Virtual function that tests if the [Component](#) meets all the requirements for a successful launch. The requirements depend on the type of [Component](#).

Returns

void

Reimplemented in [CargoDragon](#), [ComponentComposite](#), [CrewDragon](#), [Fairing](#), [FalconCore](#), [MerlinEngine](#), [VacuumMerlinEngine](#), and [RocketCapsule](#).

5.11.3 Member Data Documentation**5.11.3.1 cost**

```
double Component::cost [protected]
```

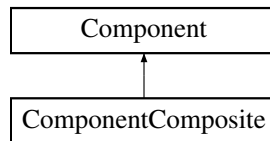
The cost of the component

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Component.h](#)
- C:/Users/labuser2/Downloads/System/System/[Component.cpp](#)

5.12 ComponentComposite Class Reference

Inheritance diagram for ComponentComposite:



Public Member Functions

- **ComponentComposite** ()
Constructor for [ComponentComposite](#) objects.
- void **simulate** ()
Starts the simulation for [ComponentComposite](#) objects.
- void **test** ()
Tests if the [ComponentComposite](#) meets all the requirements for a successful launch. The requirements depend on the type of [Component](#).
- virtual void **add** ([Component](#) *c)
Virtual function that adds a component to the rocket.
- virtual void **remove** (int pos)
Virtual function the removes a component from the rocket based on its position.
- [Component](#) * **getComponent** (int pos)
Returns a component of the rocket based on its position.
- int **getSize** ()
Returns the size of the [ComponentComposite](#) object.
- void **separate** ()
Seperates the [ComponentComposite](#) from the rocket.
- void **fireMerlin** ()
Ignites the [MerlinEngine](#) object.
- void **land** ()
Lands the rocket.

Additional Inherited Members

5.12.1 Member Function Documentation

5.12.1.1 add()

```
void ComponentComposite::add (
    Component * c ) [virtual]
```

Virtual function that adds a component to the rocket.

Parameters

| | |
|----------|---|
| <i>c</i> | Component* - the Component to be added to the rocket. |
|----------|---|

Returns

void

Reimplemented from [Component](#).

5.12.1.2 fireMerlin()

`void ComponentComposite::fireMerlin () [virtual]`
Ignites the [MerlinEngine](#) object.

Returns

void

Reimplemented from [Component](#).

5.12.1.3 getComponent()

`Component * ComponentComposite::getComponent (`
`int pos) [virtual]`

Returns a component of the rocket based on its position.

Parameters

| | |
|------------|--|
| <i>pos</i> | int - the position of the Componentet in the vector of components. |
|------------|--|

Returns

Component*

Reimplemented from [Component](#).

5.12.1.4 getSize()

`int ComponentComposite::getSize () [virtual]`
Returns the size of the [ComponentComposite](#) object.

Returns

int

Reimplemented from [Component](#).

5.12.1.5 land()

`void ComponentComposite::land () [virtual]`
Lands the rocket.

Returns

void

Reimplemented from [Component](#).

5.12.1.6 remove()

`void ComponentComposite::remove (`
`int pos) [virtual]`

Virtual function the removes a component from the rocket based on its position.

Parameters

| | |
|------------|--|
| <i>pos</i> | int - the position of the Componentet in the vector of components. |
|------------|--|

Returns

void

Reimplemented from [Component](#).**5.12.1.7 separate()**`void ComponentComposite::separate () [virtual]`Separates the [ComponentComposite](#) from the rocket.**Returns**

void

Reimplemented from [Component](#).**5.12.1.8 simulate()**`void ComponentComposite::simulate () [virtual]`Starts the simulation for [ComponentComposite](#) objects.**Returns**

void

Reimplemented from [Component](#).**5.12.1.9 test()**`void ComponentComposite::test () [virtual]`Tests if the [ComponentComposite](#) meets all the requirements for a successful launch. The requirements depend on the type of [Component](#).**Returns**

void

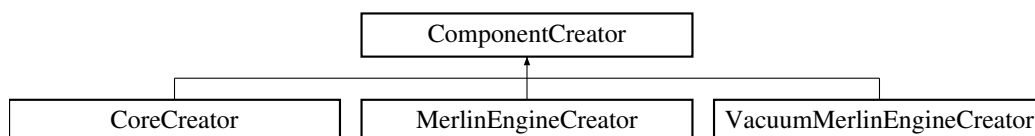
Reimplemented from [Component](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[ComponentComposite.h](#)
- C:/Users/labuser2/Downloads/System/System/[ComponentComposite.cpp](#)

5.13 ComponentCreator Class Reference

Inheritance diagram for ComponentCreator:

**Public Member Functions**

- virtual [Component](#) * [factoryMethod](#) ()=0

Virtual functon to be implemented in all the children classes. Factory method to create different types of engines.

5.13.1 Member Function Documentation

5.13.1.1 factoryMethod()

virtual [Component](#) * ComponentCreator::factoryMethod () [pure virtual]

Virtual function to be implemented in all the children classes. Factory method to create different types of engines.

Returns

Component*

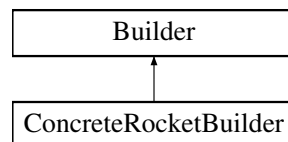
Implemented in [CoreCreator](#), [MerlinEngineCreator](#), and [VacuumMerlinEngineCreator](#).

The documentation for this class was generated from the following file:

- C:/Users/labuser2/Downloads/System/System/[ComponentCreator.h](#)

5.14 ConcreteRocketBuilder Class Reference

Inheritance diagram for ConcreteRocketBuilder:



Public Member Functions

- **ConcreteRocketBuilder** ()
Constructor for [ConcreteRocketBuilder](#) objects.
- [Component](#) * [getSpacecraft](#) ()
Returns the current spacecraft/rocket.
- [RocketCapsule](#) * [getCapsule](#) ()
Returns the current [RocketCapsule](#) object.
- void [buildFalcon9](#) ()
Builds the Falcon9 rocket. Adds the [FalconCore](#), [MerlinEngine](#) and [VacuumMerlinEngine](#) components.
- void [buildFalconHeavy](#) ()
Builds the FalconHeavy rocket. Adds the [FalconCore](#), [MerlinEngine](#) and [VacuumMerlinEngine](#) components.
- void [constructCapsule](#) (string type)
Creates a new capsule. The capsule can be either a [CrewDragon](#), [CargoDragon](#) or [Fairing](#). The capsule is constructed differently based on the capsule type.
- [Simulation](#) * [createSimulation](#) ()
Creates a new simulation based on the capsule, rocket and simulation state.

5.14.1 Member Function Documentation

5.14.1.1 buildFalcon9()

void ConcreteRocketBuilder::buildFalcon9 () [virtual]

Builds the Falcon9 rocket. Adds the [FalconCore](#), [MerlinEngine](#) and [VacuumMerlinEngine](#) components.

Returns

void

Implements [Builder](#).

5.14.1.2 buildFalconHeavy()

```
void ConcreteRocketBuilder::buildFalconHeavy ( ) [virtual]
```

Builds the FalconHeavy rocket. Adds the [FalconCore](#), [MerlinEngine](#) and [VacuumMerlinEngine](#) components.

Returns

void

Implements [Builder](#).

5.14.1.3 constructCapsule()

```
void ConcreteRocketBuilder::constructCapsule (
    string type ) [virtual]
```

Creates a new capsule. The capsule can be either a [CrewDragon](#), [CargoDragon](#) or [Fairing](#). The capsule is constructed differently based on the capsule type.

Parameters

| | |
|-------------|-------------------------------|
| <i>type</i> | string - the type of capsule. |
|-------------|-------------------------------|

Returns

void

Implements [Builder](#).

5.14.1.4 createSimulation()

```
Simulation * ConcreteRocketBuilder::createSimulation ( ) [virtual]
```

Creates a new simulation based on the capsule, rocket and simulation state.

Returns

Simulation*

Implements [Builder](#).

5.14.1.5 getCapsule()

```
RocketCapsule * ConcreteRocketBuilder::getCapsule ( )
```

Returns the current [RocketCapsule](#) object.

Returns

RocketCapsule*

5.14.1.6 getSpacecraft()

```
Component * ConcreteRocketBuilder::getSpacecraft ( ) [virtual]
```

Returns the current spacecraft/rocket.

Returns

Component*

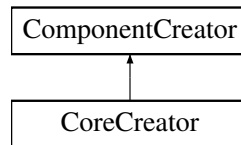
Implements [Builder](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[ConcreteRocketBuilder.h](#)
- C:/Users/labuser2/Downloads/System/System/[ConcreteRocketBuilder.cpp](#)

5.15 CoreCreator Class Reference

Inheritance diagram for CoreCreator:



Public Member Functions

- [Component](#) * [factoryMethod](#) ()
Factory method to create a new [FalconCore](#) object.

5.15.1 Member Function Documentation

5.15.1.1 factoryMethod()

[Component](#) * [CoreCreator::factoryMethod](#) () [virtual]

Factory method to create a new [FalconCore](#) object.

Returns

[Component](#)*

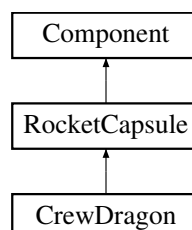
Implements [ComponentCreator](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[CoreCreator.h](#)
- C:/Users/labuser2/Downloads/System/System/[CoreCreator.cpp](#)

5.16 CrewDragon Class Reference

Inheritance diagram for CrewDragon:



Public Member Functions

- [CrewDragon](#) ([Component](#) *r)
Constructor for [CrewDragon](#) objects that takes in a rocket ([Component](#)) as a parameter and uses the [RocketCapsule](#) (parent) constructor to initialize the rocket object.
- void [simulate](#) ()
Starts the simulation for [CrewDragon](#) objects.
- void [test](#) ()
Tests if the [CrewDragon](#) meets all the requirements for a successful launch. The requirements:
- vector< string > [getPassengers](#) ()

Returns the vector of passsengers.

- void [setPassengers](#) (vector< string > p)

Sets the vector of passengers to the vector passed in as a parameter.

Additional Inherited Members

5.16.1 Constructor & Destructor Documentation

5.16.1.1 CrewDragon()

```
CrewDragon::CrewDragon (
    Component * r )
```

Constructor for [CrewDragon](#) objects that takes in a rocket ([Component](#)) as a parameter and uses the [RocketCapsule](#) (parent) constructor to initialize the rocket object.

Parameters

| | |
|----------|---|
| <i>r</i> | Component * - the rocket to which the CrewDragon object needs to be added to. |
|----------|---|

5.16.2 Member Function Documentation

5.16.2.1 getPassengers()

```
vector< string > CrewDragon::getPassengers ( )
```

Returns the vector of passsengers.

Returns

vector<string>

5.16.2.2 setPassengers()

```
void CrewDragon::setPassengers (
    vector< string > p ) [virtual]
```

Sets the vector of passengers to the vector passed in as a parameter.

Parameters

| | |
|----------|--|
| <i>p</i> | vector<string> - the vector of passengers to set to. |
|----------|--|

Returns

void

Reimplemented from [RocketCapsule](#).

5.16.2.3 simulate()

```
void CrewDragon::simulate ( ) [virtual]
```

Starts the simulation for [CrewDragon](#) objects.

Returns

void

Implements [RocketCapsule](#).**5.16.2.4 test()**

```
void CrewDragon::test ( ) [virtual]
```

Tests if the [CrewDragon](#) meets all the requirements for a successful launch. The requirements:

- the cost must be >0
- it must have a capsuleType
- it must have a rocketType

Returns

void

Implements [RocketCapsule](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[CrewDragon.h](#)
- C:/Users/labuser2/Downloads/System/System/[CrewDragon.cpp](#)

5.17 Director Class Reference**Public Member Functions**

- [Director](#) ([Builder](#) *b)
Constructor that takes in a builder object and initializes the builder variable.
- [~Director](#) ()
Destructor for [Director](#) objects.
- [Component](#) * [construct](#) ()
Constructs the rocket. [User](#) will have the choice of creating a Falcon9 or FalconHeavy rocket. [User](#) will have the option of adding a capsule to the rocket. (Capsule can be a [CrewDragon](#), [CargoDragon](#) or [Fairing](#)).
- void [constructCapsule](#) ()
Creates a new capsule of types [CrewDragon](#), [CargoDragon](#) or [Fairing](#).
- [Simulation](#) * [createSimulation](#) ()
Creates a new simulation.

5.17.1 Constructor & Destructor Documentation**5.17.1.1 Director()**

```
Director::Director (
    Builder * b )
```

Constructor that takes in a builder object and initializes the builder variable.

Parameters

| | |
|----------|--|
| <i>b</i> | Builder * - the builder associated with the Director object. |
|----------|--|

5.17.2 Member Function Documentation

5.17.2.1 construct()

`Component * Director::construct ()`

Constructs the rocket. [User](#) will have the choice of creating a Falcon9 or FalconHeavy rocket. [User](#) will have the option of adding a capsule to the rocket. (Capsule can be a [CrewDragon](#), [CargoDragon](#) or [Fairing](#)).

Returns

Component*

5.17.2.2 constructCapsule()

`void Director::constructCapsule ()`

Creates a new capsule of types [CrewDragon](#), [CargoDragon](#) or [Fairing](#).

Returns

void

5.17.2.3 createSimulation()

`Simulation * Director::createSimulation ()`

Creates a new simulation.

Returns

Simulation*

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Director.h](#)
- C:/Users/labuser2/Downloads/System/System/[Director.cpp](#)

5.18 Facade Class Reference

Public Member Functions

- **Facade** ()
Constructor for [Facade](#) objects.
- **~Facade** ()
Destructor for [Facade](#) objects.
- void **launch** ()
Tests and launches the rocket object.
- void **test** ()
Tests if the rocket object is ready to launch. Calls the [staticFireTest\(\)](#) on the simulation object.
- void **build** ()
Builds a new rocket object. Creates a new [Builder](#). Creates a new [Director](#). Creates a new [Simulation](#). Calls the [construct\(\)](#) method on the [Director](#) object to actually create the rocket. Deletes the [Director](#) object.
- void **storeSimulation** ()
Stores the simulation. Creates a [Memento](#) and stores it using [storeMemento\(\)](#).
- void **retrieveSimulation** ()
Sets the simulation variable to the [Simulation](#) stored in the [Memento](#) object.
- void **useCommNetwork** ()

- Prompts the user with various Communication Network capabilities.*
- void `separateBoosters ()`
Seperates boosters from the rocket.
- `Component *` `getRocket ()`
Returns the current rocket.
- void `fireMerlin ()`
Ignites the `MerlinEngine` objects.
- void `fireVacuumMerlin ()`
Ignites the `VacuumMerlinEngine` objects.
- void `runSimulation ()`
Starts the simulation.
- void `deliverCrew ()`
Delivers crew once in low-earth orbit.
- void `distributeSatellites ()`
Distributes satellites once in low-earth orbit.
- void `staticFireTest ()`
Tests the merlin engines and then fires them if they work.
- void `jettisonFairing ()`
Delivers `Fairing` capsule's payload (if `Fairing` is attached).
- void `printSimulation ()`
Prints a visual representation of the `Simulation` method calls.
- bool `editSimulation ()`
Makes adjustments to the store simulation.
- void `retrieveAll ()`
Retrieves all the saved simulations.

5.18.1 Member Function Documentation

5.18.1.1 `build()`

```
void Facade::build ( )
```

Builds a new rocket object. Creates a new `Builder`. Creates a new `Director`. Creates a new `Simulation`. Calls the `construct()` method on the `Director` object to actually create the rocket. Deletes the `Director` object.

Returns

void

5.18.1.2 `deliverCrew()`

```
void Facade::deliverCrew ( )
```

Delivers crew once in low-earth orbit.

Returns

void

5.18.1.3 `distributeSatellites()`

```
void Facade::distributeSatellites ( )
```

Distributes satellites once in low-earth orbit.

Returns

void

5.18.1.4 `editSimulation()`

```
bool Facade::editSimulation ( )
```

Makes adjustments to the store simulation.

Returns

void

5.18.1.5 `fireMerlin()`

```
void Facade::fireMerlin ( )
```

Ignites the [MerlinEngine](#) objects.

Returns

void.

5.18.1.6 `getRocket()`

```
Component * Facade::getRocket ( )
```

Returns the current rocket.

Returns

Component*

5.18.1.7 `jettisonFairing()`

```
void Facade::jettisonFairing ( )
```

Delivers [Fairing](#) capsule's payload (if [Fairing](#) is attached).

Returns

void

5.18.1.8 `launch()`

```
void Facade::launch ( )
```

Tests and launches the rocket object.

Returns

void

5.18.1.9 printSimulation()

```
void Facade::printSimulation ( )
```

Prints a visual representation of the [Simulation](#) method calls.

Returns

void

5.18.1.10 retrieveAll()

```
void Facade::retrieveAll ( )
```

Retrieves all the saved simulations.

Returns

void

5.18.1.11 retrieveSimulation()

```
void Facade::retrieveSimulation ( )
```

Sets the simulation variable to the [Simulation](#) stored in the [Memento](#) object.

Returns

void

5.18.1.12 runSimulation()

```
void Facade::runSimulation ( )
```

Starts the simulation.

Returns

void

5.18.1.13 separateBoosters()

```
void Facade::separateBoosters ( )
```

Seperates boosters from the rocket.

Returns

void

5.18.1.14 staticFireTest()

```
void Facade::staticFireTest ( )
```

Tests the merlin engines and then fires them if they work.

Returns

void

5.18.1.15 storeSimulation()

```
void Facade::storeSimulation ( )
```

Stores the simulation. Creates a [Memento](#) and stores it using storeMemento().

Returns

void

5.18.1.16 test()

```
void Facade::test ( )
```

Tests if the rocket object is ready to launch. Calls the [staticFireTest\(\)](#) on the simulation object.

Returns

void

5.18.1.17 useCommNetwork()

```
void Facade::useCommNetwork ( )
```

Prompts the user with various Communication Network capabilities.

Returns

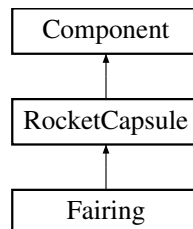
void

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Facade.h](#)
- C:/Users/labuser2/Downloads/System/System/[Facade.cpp](#)

5.19 Fairing Class Reference

Inheritance diagram for Fairing:

**Public Member Functions**

- [Fairing](#) ([Component](#) *r)
Constructor for [Fairing](#) objects that takes in a rocket ([Component](#)) as a parameter and uses the [RocketCapsule](#) (parent) constructor to initialize the rocket object.
- void [simulate](#) ()
Starts the simulation for [Fairing](#) objects.
- void [test](#) ()
Tests if the [Fairing](#) meets all the requirements for a successful launch. The requirements:
- vector< [Satellite](#) * > [getSatellites](#) ()
Returns the vector of satellites.
- void [setSatellites](#) (vector< [Satellite](#) * > s)
Sets the vector of satellites to the vector passed in as a parameter.
- [Satellite](#) * [getSatellite](#) (int id)
Returns the [Satellite](#) object with the id passed in as a parameter.

Additional Inherited Members

5.19.1 Constructor & Destructor Documentation

5.19.1.1 Fairing()

```
Fairing::Fairing (
    Component * r )
```

Constructor for [Fairing](#) objects that takes in a rocket ([Component](#)) as a parameter and uses the [RocketCapsule](#) (parent) constructor to initialize the rocket object.

Parameters

| | |
|----------|---|
| <i>r</i> | Component * - the rocket that the Fairing needs to be added to. |
|----------|---|

5.19.2 Member Function Documentation

5.19.2.1 getSatellite()

```
Satellite * Fairing::getSatellite (
    int id ) [virtual]
```

Returns the [Satellite](#) object with the id passed in as a parameter.

Parameters

| | |
|-----------|---|
| <i>id</i> | int - the id of the Satellite to be returned. |
|-----------|---|

Returns

[Satellite](#)*

Reimplemented from [RocketCapsule](#).

5.19.2.2 getSatellites()

```
vector< Satellite * > Fairing::getSatellites ( )
```

Returns the vector of satellites.

Returns

vector<[Satellite](#)*>

5.19.2.3 setSatellites()

```
void Fairing::setSatellites (
    vector< Satellite * > s ) [virtual]
```

Sets the vector of satellites to the vector passed in as a parameter.

Parameters

| | |
|----------|--|
| <i>s</i> | vector< Satellite *> - the vector of Satellite objects to be set to. |
|----------|--|

Reimplemented from [RocketCapsule](#).

5.19.2.4 simulate()

```
void Fairing::simulate ( ) [virtual]
```

Starts the simulation for [Fairing](#) objects.

Returns

void

Implements [RocketCapsule](#).

5.19.2.5 test()

```
void Fairing::test ( ) [virtual]
```

Tests if the [Fairing](#) meets all the requirements for a successful launch. The requirements:

- the cost must be >0
- it must have a capsuleType
- it must have a rocketType

Returns

void

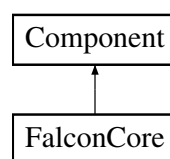
Implements [RocketCapsule](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Fairing.h](#)
- C:/Users/labuser2/Downloads/System/System/[Fairing.cpp](#)

5.20 FalconCore Class Reference

Inheritance diagram for FalconCore:



Public Member Functions

- **FalconCore** ()
Constructor for [FalconCore](#) objects.
- void **simulate** ()
Starts the simulation for [FalconCore](#) objects.
- void **test** ()
Tests if the [FalconCore](#) meets all the requirements for a successful launch. The requirements:
- void **separate** ()
Outputs that the [FalconCore](#) has been seperated from the rocket.
- void **land** ()
Outputs that the rocket has landed.

Additional Inherited Members

5.20.1 Member Function Documentation

5.20.1.1 `land()`

```
void FalconCore::land ( ) [virtual]
```

Outputs that the rocket has landed.

Returns

void

Reimplemented from [Component](#).

5.20.1.2 `separate()`

```
void FalconCore::separate ( ) [virtual]
```

Outputs that the [FalconCore](#) has been seperated from the rocket.

Returns

void

Reimplemented from [Component](#).

5.20.1.3 `simulate()`

```
void FalconCore::simulate ( ) [virtual]
```

Starts the simulation for [FalconCore](#) objects.

Returns

void

Reimplemented from [Component](#).

5.20.1.4 `test()`

```
void FalconCore::test ( ) [virtual]
```

Tests if the [FalconCore](#) meets all the requirements for a successful launch. The requirements:

- the cost must be >0
- it must have a capsuleType
- it must have a rocketType

Returns

void

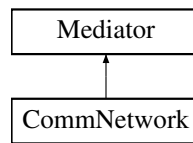
Reimplemented from [Component](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[FalconCore.h](#)
- C:/Users/labuser2/Downloads/System/System/[FalconCore.cpp](#)

5.21 Mediator Class Reference

Inheritance diagram for Mediator:



Public Member Functions

- virtual void [notify](#) (int sender)=0
Pure virtual function to be implemented in all children. Notifies all the satellites (colleagues) if the state of the one of the satellites have changed.
- virtual void [sendMessage](#) (int sender, int receiver, string msg)=0
Sends a string message to a particular satellite.

5.21.1 Member Function Documentation

5.21.1.1 notify()

```
virtual void Mediator::notify (
    int sender ) [pure virtual]
```

Pure virtual function to be implemented in all children. Notifies all the satellites (colleagues) if the state of the one of the satellites have changed.

Parameters

| | |
|---------------------------|--|
| colleague | Satellite* - the Satellite object that has changed states. |
|---------------------------|--|

Returns

void

Pure virtual function to be implemented in all children. Notifies all the satellites (colleagues) if the state of the one of the satellites have changed.

Parameters

| | |
|---------------------------|--|
| colleague | Satellite* - the Satellite object that has changed states. |
|---------------------------|--|

Returns

void

Implemented in [CommNetwork](#).

5.21.1.2 sendMessage()

```
virtual void Mediator::sendMessage (
    int sender,
    int receiver,
    string msg ) [pure virtual]
```

Sends a string message to a particular satellite.

Parameters

| | |
|-----------------|------------------------|
| <i>sender</i> | The ID of the sender. |
| <i>receiver</i> | The ID of the receiver |
| <i>msg</i> | The message to send |

Returns

void

Implemented in [CommNetwork](#).

The documentation for this class was generated from the following file:

- C:/Users/labuser2/Downloads/System/System/[Mediator.h](#)

5.22 Memento Class Reference

Public Member Functions

- [SimulationState](#) * [getState](#) ()
Returns the current state of the simulation.
- void [setState](#) ([SimulationState](#) *c)
Sets the simulationState the the object passed in as a paramter.
- ~**Memento** ()
Destructor for the [Memento](#) class.

5.22.1 Member Function Documentation

5.22.1.1 [getState\(\)](#)

```
SimulationState * Memento::getState ( )
```

Returns the current state of the simulation.

Returns

[SimulationState](#)*

5.22.1.2 [setState\(\)](#)

```
void Memento::setState (
    SimulationState * c )
```

Sets the simulationState the the object passed in as a paramter.

Parameters

| | |
|----------|---|
| <i>c</i> | SimulationState * - the SimulationState to be set to. |
|----------|---|

Returns

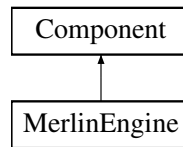
void

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Memento.h](#)
- C:/Users/labuser2/Downloads/System/System/[Memento.cpp](#)

5.23 MerlinEngine Class Reference

Inheritance diagram for MerlinEngine:



Public Member Functions

- **MerlinEngine** ()
Constructor for [MerlinEngine](#) objects. Uses the [Component](#) (parent) constructor to initialize the object.
- void **simulate** ()
Starts the simulation for [MerlinEngine](#) objects.
- void **test** ()
Tests if the [MerlinEngine](#) meets all the requirements for a successful launch. The requirements:
- void **fireMerlin** ()
Outputs that a [MerlinEngine](#) has been ignited.

Additional Inherited Members

5.23.1 Member Function Documentation

5.23.1.1 fireMerlin()

```
void MerlinEngine::fireMerlin ( ) [virtual]
```

Outputs that a [MerlinEngine](#) has been ignited.

Returns

void

Reimplemented from [Component](#).

5.23.1.2 simulate()

```
void MerlinEngine::simulate ( ) [virtual]
```

Starts the simulation for [MerlinEngine](#) objects.

Returns

void

Reimplemented from [Component](#).

5.23.1.3 test()

```
void MerlinEngine::test ( ) [virtual]
```

Tests if the [MerlinEngine](#) meets all the requirements for a successful launch. The requirements:

- the cost must be >0
- it must have a capsuleType
- it must have a rocketType

Returns

void

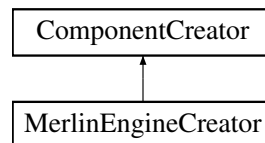
Reimplemented from [Component](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[MerlinEngine.h](#)
- C:/Users/labuser2/Downloads/System/System/[MerlinEngine.cpp](#)

5.24 MerlinEngineCreator Class Reference

Inheritance diagram for MerlinEngineCreator:



Public Member Functions

- [Component](#) * [factoryMethod](#) ()
Factory method to create a [MerlinEngine](#).

5.24.1 Member Function Documentation

5.24.1.1 factoryMethod()

```
Component * MerlinEngineCreator::factoryMethod ( ) [virtual]
```

Factory method to create a [MerlinEngine](#).

Returns

Component*

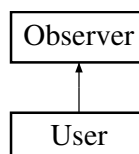
Implements [ComponentCreator](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[MerlinEngineCreator.h](#)
- C:/Users/labuser2/Downloads/System/System/[MerlinEngineCreator.cpp](#)

5.25 Observer Class Reference

Inheritance diagram for Observer:



Public Member Functions

- virtual void [update](#) (int satelliteID, string status)=0
Pure virtual function to be implemented in all the children classes. Updates the state of a satellite the class is currently observing.

5.25.1 Member Function Documentation

5.25.1.1 update()

```
virtual void Observer::update (
    int satelliteID,
    string status ) [pure virtual]
```

Pure virtual function to be implemented in all the children classes. Updates the state of a satellite the class is currently observing.

Returns

void

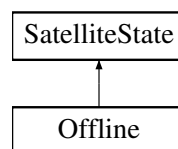
Implemented in [User](#).

The documentation for this class was generated from the following file:

- C:/Users/labuser2/Downloads/System/System/[Observer.h](#)

5.26 Offline Class Reference

Inheritance diagram for Offline:



Public Member Functions

- **Offline** ()
Constructor for [Offline](#) objects.
- string [getType](#) ()
Returns the type of state the satellite is currently in ([Offline](#)).
- [SatelliteState](#) * [handleChange](#) ()
Handles a change in state - sets the current state of the satellite to null.

5.26.1 Member Function Documentation

5.26.1.1 getType()

```
string Offline::getType ( ) [virtual]
```

Returns the type of state the satellite is currently in ([Offline](#)).

Returns

string

Implements [SatelliteState](#).

5.26.1.2 handleChange()

```
SatelliteState * Offline::handleChange ( ) [virtual]
```

Handles a change in state - sets the current state of the satellite to null.

Returns

SatelliteState*

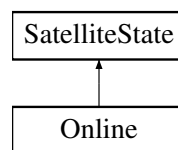
Implements [SatelliteState](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Offline.h](#)
- C:/Users/labuser2/Downloads/System/System/[Offline.cpp](#)

5.27 Online Class Reference

Inheritance diagram for Online:



Public Member Functions

- **Online ()**
Constructor for [Online](#) objects.
- string [getType \(\)](#)
Returns the type of state the satellite is currently in ([Online](#)).
- [SatelliteState * handleChange \(\)](#)
Handles a change in state - sets the current state of the satellite to '[Broadcasting](#)'.

5.27.1 Member Function Documentation

5.27.1.1 getType()

```
string Online::getType ( ) [virtual]
```

Returns the type of state the satellite is currently in ([Online](#)).

Returns

string

Implements [SatelliteState](#).

5.27.1.2 handleChange()

```
SatelliteState * Online::handleChange ( ) [virtual]
```

Handles a change in state - sets the current state of the satellite to '[Broadcasting](#)'.

Returns

SatelliteState*

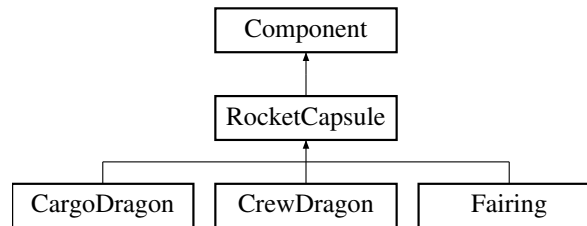
Implements [SatelliteState](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Online.h](#)
- C:/Users/labuser2/Downloads/System/System/[Online.cpp](#)

5.28 RocketCapsule Class Reference

Inheritance diagram for RocketCapsule:



Public Member Functions

- **RocketCapsule** ([Component](#) *r)
Constructor for [RocketCapsule](#) objects. Sets the rocket variable to the [Component](#) sent in as a parameter. Sets the state to [CapsuleOffline](#).
- virtual void [simulate](#) ()=0
Pure virtual function to be implemented in all the children classes. Starts the simulation for [RocketCapsule](#) objects.
- virtual void [test](#) ()=0
Tests if the [RocketCapsule](#) meets all the requirements for a successful launch. The requirements:
- void [addCapsule](#) ([Component](#) *r)
Adds a capsule to the rocket.
- void [requestStateChange](#) ()
Requests a state change of the [RocketCapsule](#). Calls the [handleChange](#) method on the [CapsuleState](#) object.
- void [setState](#) ([CapsuleState](#) *s)
Sets the state of the [RocketCapsule](#) to the [CapsuleState](#) passed in as a parameter.
- double [getPayloadWeight](#) ()
Getter function to return the [payloadWeight](#) of the [RocketCapsule](#).
- void [setPayloadWeight](#) (double pw)
Setter function to set the [payloadWeight](#) of the [RocketCapsule](#) to the value passed in as a parameter.
- virtual void [setPassengers](#) (vector< string > p)
Virtual setter function to set the vector of passengers in a [CrewDragon RocketCapsule](#). Only implemented in the [CrewDragon](#) class.
- virtual void [setSatellites](#) (vector< [Satellite](#) * > s)
Virtual setter function to set the vector of satellites in a [Fairing RocketCapsule](#). Only implemented in the [Fairing](#) class.
- virtual [Satellite](#) * [getSatellite](#) (int id)
Get a specific [Satellite](#) object on board the [Fairing](#).
- [CapsuleState](#) * [getState](#) ()
Get the [CapsuleState](#) object.

Protected Attributes

- string [capsuleType](#)
- [CapsuleState](#) * [state](#)

5.28.1 Member Function Documentation

5.28.1.1 addCapsule()

```
void RocketCapsule::addCapsule (
    Component * r )
```

Adds a capsule to the rocket.

Parameters

| | |
|----------|---|
| <i>r</i> | Component* - the rocket to be added to. |
|----------|---|

Returns

void

5.28.1.2 getPayloadWeight()

```
double RocketCapsule::getPayloadWeight ( )
```

Getter function to return the payloadWeight of the [RocketCapsule](#).

Returns

double

5.28.1.3 getSatellite()

```
Satellite * RocketCapsule::getSatellite (
    int id ) [virtual]
```

Get a specific [Satellite](#) object on board the [Fairing](#).

Returns

Satellite*

Reimplemented in [Fairing](#).

5.28.1.4 getState()

```
CapsuleState * RocketCapsule::getState ( )
```

Get the [CapsuleState](#) object.

Returns

CapsuleState*

5.28.1.5 requestStateChange()

```
void RocketCapsule::requestStateChange ( )
```

Requests a state change of the [RocketCapsule](#). Calls the handleChange method on the [CapsuleState](#) object.

Returns

void

5.28.1.6 setPassengers()

```
virtual void RocketCapsule::setPassengers (
    vector< string > p ) [inline], [virtual]
```

Virtual setter function to set the vector of passengers in a [CrewDragon RocketCapsule](#). Only implemented in the [CrewDragon](#) class.

Parameters

| | |
|----------|--|
| <i>p</i> | vector<string> - the string vector containing the names of all the passengers. |
|----------|--|

Returns

void

Reimplemented in [CrewDragon](#).

5.28.1.7 setPayloadWeight()

```
void RocketCapsule::setPayloadWeight (
    double pw )
```

Setter function to set the payloadWeight of the [RocketCapsule](#) to the value passed in as a parameter.

Parameters

| | |
|-----------|----------------------------------|
| <i>pw</i> | double - the value to be set to. |
|-----------|----------------------------------|

5.28.1.8 setSatellites()

```
virtual void RocketCapsule::setSatellites (
    vector< Satellite * > s ) [inline], [virtual]
```

Virtual setter function to set the vector of satellites in a [Fairing RocketCapsule](#). Only implemented in the [Fairing](#) class.

Parameters

| | |
|----------|---|
| <i>s</i> | vector<Satellite*> - vector of all the Satellite objects in the Fairing . |
|----------|---|

Returns

void

Reimplemented in [Fairing](#).

5.28.1.9 setState()

```
void RocketCapsule::setState (
    CapsuleState * s )
```

Sets the state of the [RocketCapsule](#) to the [CapsuleState](#) passed in as a parameter.

Parameters

| | |
|----------|---|
| <i>s</i> | CapsuleState* - the state to be set to. |
|----------|---|

Returns

void

5.28.1.10 simulate()

```
void RocketCapsule::simulate ( ) [pure virtual]
```

Pure virtual function to be implemented in all the children classes. Starts the simulation for [RocketCapsule](#) objects.

Returns

void

Reimplemented from [Component](#).

Implemented in [CargoDragon](#), [CrewDragon](#), and [Fairing](#).

5.28.1.11 test()

```
void RocketCapsule::test ( ) [pure virtual]
```

Tests if the [RocketCapsule](#) meets all the requirements for a successful launch. The requirements:

- the cost must be >0
- it must have a capsuleType
- it must have a rocketType

Returns

void

Reimplemented from [Component](#).

Implemented in [CargoDragon](#), [CrewDragon](#), and [Fairing](#).

5.28.2 Member Data Documentation**5.28.2.1 capsuleType**

```
string RocketCapsule::capsuleType [protected]
```

The type of capsule ([CrewDragon](#), [CargoDragon](#), [Fairing](#)).

5.28.2.2 state

```
CapsuleState* RocketCapsule::state [protected]
```

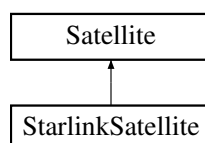
The state of the capsule (Docked, Arrriving, Departing, [Offline](#)).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[RocketCapsule.h](#)
- C:/Users/labuser2/Downloads/System/System/[RocketCapsule.cpp](#)

5.29 Satellite Class Reference

Inheritance diagram for Satellite:



Public Member Functions

- [Satellite](#) (int ID)
Constructor for [Satellite](#) objects. Sets the ID to the value sent in as a parameter.
- [~Satellite](#) ()
Destructor for [Satellite](#) objects.
- void [changed](#) ()
Notifies the other satellites that the state of the current satellite has changed by calling the [notify\(\)](#) method on the mediator object.
- int [getID](#) ()
Getter method that returns the ID of the [Satellite](#) object.
- void [sendMessage](#) (int id, string msg)
Sends a string message to a particular satellite, based on ID.
- void [receiveMessage](#) (int id, string msg)
Prints out the state and ID of the changed colleague.
- void [setMediator](#) ([Mediator](#) *m)
Setter function to set the mediator to the [Mediator](#) object passed in as a parameter.
- [Mediator](#) * [getMediator](#) ()
- void [requestStateChange](#) ()
Changes the state of the satellite by calling [setState\(\)](#). Notifies the mediators by calling [changed\(\)](#). Notifies the observers by calling [notify\(\)](#).
- void [attach](#) ([Observer](#) *o)
Registers an observer with the [Satellite](#). Adds the [Observer](#) sent in as a parameter to the observerList.
- void [detach](#) ([Observer](#) *o)
Deregisters an observer from the [Satellite](#). Removes the [Observer](#) sent in as a parameter from the observerList.
- void [notify](#) ()
Notifies all the observers of the state change.
- virtual [SatelliteState](#) * [getState](#) ()
Virtual function that returns the state of the [Satellite](#).
- void [setState](#) ([SatelliteState](#) *s)
Sets the state of the [Satellite](#) to the [SatelliteState](#) sent in as a parameter.
- virtual [Satellite](#) * [clone](#) (int id)
Virtual function that clones a [Satellite](#) object. Returns NULL.

Protected Attributes

- [SatelliteState](#) * [satelliteState](#)
- [Mediator](#) * [mediator](#)
- vector< [Observer](#) * > [observerList](#)
- int ID

5.29.1 Constructor & Destructor Documentation

5.29.1.1 [Satellite](#)()

```
Satellite::Satellite (
    int ID )
```

Constructor for [Satellite](#) objects. Sets the ID to the value sent in as a parameter.

Parameters

| | |
|-----------|--------------------------------|
| <i>ID</i> | int - the id of the satellite. |
|-----------|--------------------------------|

5.29.2 Member Function Documentation

5.29.2.1 attach()

```
void Satellite::attach (
    Observer * o )
```

Registers an observer with the [Satellite](#). Adds the [Observer](#) sent in as a parameter to the observerList.

Parameters

| | |
|----------|--|
| <i>o</i> | Observer* - the Observer to be registered. |
|----------|--|

Returns

void

5.29.2.2 changed()

```
void Satellite::changed ( )
```

Notifies the other satellites that the state of the current satellite has changed by calling the [notify\(\)](#) method on the mediator object.

Returns

void

5.29.2.3 clone()

```
virtual Satellite * Satellite::clone (
    int id ) [inline], [virtual]
```

Virtual function that clones a [Satellite](#) object. Returns NULL.

Returns

Satellite*

Reimplemented in [StarlinkSatellite](#).

5.29.2.4 detach()

```
void Satellite::detach (
    Observer * o )
```

Deregisters an observer from the [Satellite](#). Removes the [Observer](#) sent in as a paramter from the observerList.

Parameters

| | |
|----------|--|
| <i>o</i> | Observer* - the Observer to be deregistered. |
|----------|--|

Returns

void

5.29.2.5 getID()

```
int Satellite::getID ( )
```

Getter method that returns the ID of the [Satellite](#) object.

Returns

int

5.29.2.6 getState()

```
SatelliteState * Satellite::getState ( ) [virtual]
```

Virtual function that returns the state of the [Satellite](#).

Returns

SatelliteState*

Reimplemented in [StarlinkSatellite](#).

5.29.2.7 notify()

```
void Satellite::notify ( )
```

Notifies all the observers of the state change.

Returns

void

5.29.2.8 receiveMessage()

```
void Satellite::receiveMessage (
    int id,
    string msg )
```

Prints out the state and ID of the changed colleague.

Parameters

| | |
|------------|--|
| <i>msg</i> | string - the message that needs to be printed. |
|------------|--|

Returns

void

5.29.2.9 requestStateChange()

```
void Satellite::requestStateChange ( )
```

Changes the state of the satellite by calling [setState\(\)](#). Notifies the mediators by calling [changed\(\)](#). Notifies the observers by calling [notify\(\)](#).

Returns

void

5.29.2.10 sendMessage()

```
void Satellite::sendMessage (
    int id,
    string msg )
```

Sends a string message to a particular satellite, based on ID.

Parameters

| | |
|------------|--|
| <i>id</i> | int - The ID of the satellite. |
| <i>msg</i> | string - The message that needs to be printed. |

Returns

void

5.29.2.11 setMediator()

```
void Satellite::setMediator (
    Mediator * m )
```

Setter function to set the mediator to the [Mediator](#) object passed in as a parameter.

Parameters

| | |
|----------|---|
| <i>m</i> | Mediator* - the Mediator object to be set to. |
|----------|---|

Returns

void

5.29.2.12 setState()

```
void Satellite::setState (
    SatelliteState * s )
```

Sets the state of the [Satellite](#) to the [SatelliteState](#) sent in as a parameter.

Parameters

| | |
|----------|---|
| <i>s</i> | SatelliteState* - the state to be set to. |
|----------|---|

Returns

void

5.29.3 Member Data Documentation

5.29.3.1 ID

```
int Satellite::ID [protected]
```

The integer used to uniquely identify the satellite.

5.29.3.2 mediator

```
Mediator* Satellite::mediator [protected]
```

The mediator object that controls the communication between satellites.

5.29.3.3 observerList

```
vector<Observer*> Satellite::observerList [protected]
```

The list of observers that are observing and monitoring the satellite.

5.29.3.4 satelliteState

```
SatelliteState* Satellite::satelliteState [protected]
```

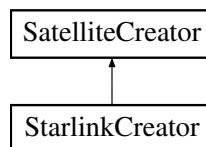
The state of the [Satellite](#)

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Satellite.h](#)
- C:/Users/labuser2/Downloads/System/System/[Satellite.cpp](#)

5.30 SatelliteCreator Class Reference

Inheritance diagram for SatelliteCreator:



Public Member Functions

- **SatelliteCreator ()**
Constructor for the [SatelliteCreator](#) objects. Sets count to 0.
- virtual [Satellite](#) * **factoryMethod ()**=0
Pure virtual function to be implemented in children classes. Factory method to create [Satellite](#) objects.
- virtual void **setIDCount** (int id)=0
Pure virtual function to be implemented in children classes. Sets the count variable to the integer passed in as a parameter.

5.30.1 Member Function Documentation

5.30.1.1 factoryMethod()

```
Satellite * SatelliteCreator::factoryMethod ( ) [pure virtual]
```

Pure virtual function to be implemented in children classes. Factory method to create [Satellite](#) objects.

Returns

[Satellite](#)*

Implemented in [StarlinkCreator](#).

5.30.1.2 setIDCount()

```
virtual void SatelliteCreator::setIDCount (
    int id ) [pure virtual]
```

Pure virtual function to be implemented in children classes. Sets the count variable to the integer passed in as a parameter.

Parameters

| | |
|-----------|--------------------------------|
| <i>id</i> | int - the number to be set to. |
|-----------|--------------------------------|

Returns

void

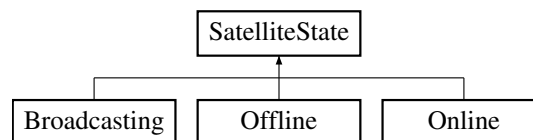
Implemented in [StarlinkCreator](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[SatelliteCreator.h](#)
- C:/Users/labuser2/Downloads/System/System/[SatelliteCreator.cpp](#)

5.31 SatelliteState Class Reference

Inheritance diagram for SatelliteState:



Public Member Functions

- virtual string [getType](#) ()=0
Pure virtual function to be implemented in children classes. Returns the name of the current state of the [Satellite](#).
- virtual [SatelliteState](#) * [handleChange](#) ()=0
Pure virtual function to be implemented in children classes. Handles a change in state.

5.31.1 Member Function Documentation

5.31.1.1 getType()

```
virtual string SatelliteState::getType ( ) [pure virtual]
```

Pure virtual function to be implemented in children classes. Returns the name of the current state of the [Satellite](#).

Returns

string

Implemented in [Broadcasting](#), [Offline](#), and [Online](#).

5.31.1.2 handleChange()

```
virtual SatelliteState * SatelliteState::handleChange ( ) [pure virtual]
```

Pure virtual function to be implemented in children classes. Handles a change in state.

Returns

[SatelliteState](#)*

Implemented in [Broadcasting](#), [Offline](#), and [Online](#).

The documentation for this class was generated from the following file:

- C:/Users/labuser2/Downloads/System/System/[SatelliteState.h](#)

5.32 Simulation Class Reference

Public Member Functions

- [Simulation](#) ([RocketCapsule](#) *c, [Component](#) *r, [SimulationState](#) *s)
Constructor for [Simulation](#) objects. Sets the simulationState, rocket and capsule variables.
- [RocketCapsule](#) * [getCapsule](#) ()
- [Memento](#) * [createMemento](#) ()
Creates a [Memento](#) object and sets the state of the memento to the current simulationState.
- void [restoreMemento](#) ([Memento](#) *m)
Sets the simulationState to the state of the memento.
- void [staticFireTest](#) ()
Adds a call to the simulationState and calls the test() method on the rocket.
- void [launch](#) ()
Launches the rocket. IF FALCON9 ROCKET- – Stage 1: single falcon 9 core with 9 Merlin engines – Stage 2: single vacuum Merlin engine IF FALCONHEAVY ROCKET- – Stage 1: 3 Falcon Heavy cores with 27 Merlin engines – Stage 2: single Merlin engine.
- void [printSimulation](#) ()
Pure virtual method to be implemented in all children classes. Tweaks the simulation on the rocket to represent a more realistic example of a real-world rocket simulation.
- void [jettisonFairing](#) ()
Delivers [Fairing](#) capsule's payload (if [Fairing](#) is attached).
- void [separateBoosters](#) ()
Seperates boosters from the rocket.
- void [distributeSatellites](#) ()
Distributes satellites once in low-earth orbit.
- void [deliverCrew](#) ()
Delivers crew once in low-earth orbit.
- void [sendMessage](#) (int sender, int reciever, string message)
Sends a string message from the receiving satellite to a receiving satellite.
- void [runSimulation](#) ()
Runs various launch methods from the simulationState consecutively in order to simulate launch event.
- [SimulationState](#) * [getState](#) ()
Returns the state of te [Simulation](#).
- void [fireMerlin](#) ()
Outputs that a [MerlinEngine](#) has been ignited.
- void [landBoosters](#) ()
Outputs that the boosters have landed.
- void [fireVacuumMerlin](#) ()
Outputs that a [VacuumMerlinEngine](#) has been ignited.
- void [changeSatelliteState](#) (int id, [SatelliteState](#) *state)

- Changes the state of the [Satellite](#) with an id of the int passed in as a parameter to the state passed in as a parameter.
- void [addCall](#) (string c)
Adds a call to vector of methodCalls.
- bool [containsCall](#) (string c)
Checks if the methodCall vector contains the string passed in as a parameter. Returns true if it contains it. Returns false if it doesn't contain it.
- void [updateSimulationState](#) ()
Updates the simulation state to the most recent method calls.
- void [swapStage](#) (int pos_1, int pos_2)
Swaps between the indexes of the 2 method calls sent in as parameters.
- void [removeStage](#) (int pos)
Removes the method call in the index of the number sent in as a parameter.
- int [getSimulationSize](#) ()
Returns the size of the simulation - the number of method calls/ stages in the simulation.

5.32.1 Constructor & Destructor Documentation

5.32.1.1 Simulation()

```
Simulation::Simulation (
    RocketCapsule * c,
    Component * r,
    SimulationState * s )
```

Constructor for [Simulation](#) objects. Sets the simulationState, rocket and capsule variables.

Parameters

| | |
|----------|--|
| <i>c</i> | Component* - The capsule on the rocket on which the Simulation is performed. |
| <i>r</i> | Component* - The rocket on which the Simulation is performed. |
| <i>s</i> | SimulationState* - The state of the Simulation . |

5.32.2 Member Function Documentation

5.32.2.1 addCall()

```
void Simulation::addCall (
    string c )
```

Adds a call to vector of methodCalls.

Parameters

| | |
|----------|--|
| <i>c</i> | string - the method call to add to the vector. |
|----------|--|

Returns

void

5.32.2.2 changeSatelliteState()

```
void Simulation::changeSatelliteState (
```

```
int id,  
SatelliteState * state )
```

Changes the state of the [Satellite](#) with an id of the int passed in as a parameter to the state passed in as a parameter.

Parameters

| | |
|--------------|---|
| <i>id</i> | int - the id of the Satellite of which we need to change the state. |
| <i>state</i> | SatelliteState* - the state to change to. |

5.32.2.3 containsCall()

```
bool Simulation::containsCall (  
    string c )
```

Checks if the methodCall vector contains the string passed in as a parameter. Returns true if it contains it. Returns false if it doesn't contain it.

Parameters

| | |
|----------|--|
| <i>c</i> | string - the method call to check for. |
|----------|--|

Returns

bool

5.32.2.4 createMemento()

```
Memento * Simulation::createMemento ( )
```

Creates a [Memento](#) object and sets the state of the memento to the current simulationState.

Returns

Memento*

5.32.2.5 deliverCrew()

```
void Simulation::deliverCrew ( )
```

Delivers crew once in low-earth orbit.

Returns

void

5.32.2.6 distributeSatellites()

```
void Simulation::distributeSatellites ( )
```

Distributes satellites once in low-earth orbit.

Returns

void

5.32.2.7 fireMerlin()

```
void Simulation::fireMerlin ( )
```

Outputs that a [MerlinEngine](#) has been ignited.

Returns

void

5.32.2.8 fireVacuumMerlin()

```
void Simulation::fireVacuumMerlin ( )
```

Outputs that a [VacuumMerlinEngine](#) has been ignited.

Returns

void

5.32.2.9 getSimulationSize()

```
int Simulation::getSimulationSize ( )
```

Returns the size of the simulation - the number of method calls/ stages in the simulation.

Returns

int

5.32.2.10 getState()

```
SimulationState * Simulation::getState ( )
```

Returns the state of the [Simulation](#).

Returns

SimulationState*

5.32.2.11 jettisonFairing()

```
void Simulation::jettisonFairing ( )
```

Delivers [Fairing](#) capsule's payload (if [Fairing](#) is attached).

Returns

void

5.32.2.12 landBoosters()

```
void Simulation::landBoosters ( )
```

Outputs that the boosters have landed.

Returns

void

5.32.2.13 launch()

```
void Simulation::launch ( )
```

Launches the rocket. IF FALCON9 ROCKET- – Stage 1: single falcon 9 core with 9 Merlin engines – Stage 2: single vacuum Merlin engine IF FALCONHEAVY ROCKET- – Stage 1: 3 Falcon Heavy cores with 27 Merlin engines – Stage 2: single Merlin engine.

Returns

void

5.32.2.14 printSimulation()

```
void Simulation::printSimulation ( )
```

Pure virtual method to be implemented in all children classes. Tweaks the simulation on the rocket to represent a more realistic example of a real-world rocket simulation.

Returns

void

Prints a visual representation of the [Simulation](#) method calls.

Returns

void

5.32.2.15 removeStage()

```
void Simulation::removeStage (
    int pos )
```

Removes the method call in the index of the number sent in as a parameter.

Parameters

| | |
|------------|--|
| <i>pos</i> | int - the position of the method call to be removed. |
|------------|--|

Returns

void

5.32.2.16 restoreMemento()

```
void Simulation::restoreMemento (
    Memento * m )
```

Sets the simulationState to the state of the memento.

Parameters

| | |
|----------|---|
| <i>m</i> | Memento* - the Memento object storing the current simulation. |
|----------|---|

Returns

void

5.32.2.17 runSimulation()

```
void Simulation::runSimulation ( )
```

Runs various launch methods from the simulationState consecutively in order to simulate launch event.

Returns

void

5.32.2.18 sendMessage()

```
void Simulation::sendMessage (
    int sender,
    int reciever,
    string message )
```

Sends a string message from the receiving satellite to a receiving satellite.

Returns

void

5.32.2.19 separateBoosters()

```
void Simulation::separateBoosters ( )
```

Seperates boosters from the rocket.

Returns

void

5.32.2.20 staticFireTest()

```
void Simulation::staticFireTest ( )
```

Adds a call to the simulationState and calls the test() method on the rocket.

Returns

void

5.32.2.21 swapStage()

```
void Simulation::swapStage (
    int pos_1,
    int pos_2 )
```

Swaps between the indexes of the 2 method calls sent in as parameters.

Parameters

| | |
|---------------------|-------------------------|
| <i>pos_1</i> | int - current position. |
| <i>pos2↔ _2</i> | int - current position. |

Returns

void

5.32.2.22 updateSimulationState()

```
void Simulation::updateSimulationState ( )
```

Updates the simulation state to the most recent method calls.

Returns

void

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[Simulation.h](#)
- C:/Users/labuser2/Downloads/System/System/[Simulation.cpp](#)

5.33 SimulationState Class Reference

Public Member Functions

- **SimulationState ()**
Constructor for [SimulationState](#) objects.
- string [getCapsuleType \(\)](#)
Returns the capsule type of the rocket ([CrewDragon](#), [CargoDragon](#), [Fairing](#)).
- string [getRocketType \(\)](#)
Returns the rocket type of the rocket ([Falcon9](#), [FalconHeavy](#)).
- double [getPayloadWeight \(\)](#)
Returns the payloadWeight of the capsule.
- vector< [Satellite](#) * > [getSatellites \(\)](#)
Returns the vector of satellites on the rocket (if capsuleType==[Fairing](#)).
- vector< string > [getPassengers \(\)](#)
Returns the vector of passengers on the rocket (if capsuleType==[CrewDragon](#)).
- vector< string > [getMethodCalls \(\)](#)
Returns the vector of methodCalls on the rocket.
- void [setCapsuleType](#) (string s)
Sets the capsuleType to the string passed in as a parameter.
- void [setRocketType](#) (string s)
- void [setPayloadWeight](#) (double d)
Sets the payloadWeight to the string passed in as a parameter.
- void [setSatellites](#) (vector< [Satellite](#) * > s)
Sets the vector of Satellites to the vector passed in as a parameter.
- void [setPassengers](#) (vector< string > p)
Sets the vector of passengers (strings) to the vector passed in as a parameter.
- void [setMethodCalls](#) (vector< string > c)
Sets the vector of method calls (strings) to the vector passed in as a parameter.

5.33.1 Member Function Documentation

5.33.1.1 getCapsuleType()

```
string SimulationState::getCapsuleType ( )
```

Returns the capsule type of the rocket ([CrewDragon](#), [CargoDragon](#), [Fairing](#)).

Returns

string

5.33.1.2 getMethodCalls()

```
vector< string > SimulationState::getMethodCalls ( )
```

Returns the vector of methodCalls on the rocket.

Returns

vector<string>

5.33.1.3 getPassengers()

```
vector< string > SimulationState::getPassengers ( )
```

Returns the vector of passengers on the rocket (if capsuleType==[CrewDragon](#)).

Returns

vector<string>

5.33.1.4 getPayloadWeight()

```
double SimulationState::getPayloadWeight ( )
```

Returns the payloadWeight of the capsule.

Returns

double

5.33.1.5 getRocketType()

```
string SimulationState::getRocketType ( )
```

Returns the rocket type of the rocket ([Falcon9](#), [FalconHeavy](#)).

Returns

string

5.33.1.6 getSatellites()

```
vector< Satellite * > SimulationState::getSatellites ( )
```

Returns the vector of satellites on the rocket (if capsuleType==[Fairing](#)).

Returns

vector<Satellite*>

5.33.1.7 setCapsuleType()

```
void SimulationState::setCapsuleType (
    string s )
```

Sets the capsuleType to the string passed in as a parameter.

Parameters

| | |
|----------|-------------------------------------|
| <i>s</i> | string - the capsuleType to set to. |
|----------|-------------------------------------|

Returns

void

5.33.1.8 setMethodCalls()

```
void SimulationState::setMethodCalls (
    vector< string > c )
```

Sets the vector of method calls (strings) to the vector passed in as a parameter.

Parameters

| | |
|----------|--|
| <i>c</i> | vector<string> - the vector of string to set to. |
|----------|--|

Returns

void

5.33.1.9 setPassengers()

```
void SimulationState::setPassengers (
    vector< string > p )
```

Sets the vector of passengers (strings) to the vector passed in as a parameter.

Parameters

| | |
|----------|---|
| <i>p</i> | vector<string> - the vector of strings to set to. |
|----------|---|

Returns

void

5.33.1.10 setPayloadWeight()

```
void SimulationState::setPayloadWeight (
    double d )
```

Sets the payloadWeight to the string passed in as a parameter.

Parameters

| | |
|----------|-------------------------------|
| <i>d</i> | double - the value to set to. |
|----------|-------------------------------|

Returns

void

5.33.1.11 setRocketType()

```
void SimulationState::setRocketType (
    string s )
```

Sets the rocketType to the string passed in as a parameter.

Parameters

| | |
|---|------------------------------------|
| s | string - the rocketType to set to. |
|---|------------------------------------|

Returns

void

5.33.1.12 setSatellites()

```
void SimulationState::setSatellites (
    vector< Satellite * > s )
```

Sets the vector of Satellites to the vector passed in as a parameter.

Parameters

| | |
|---|--|
| s | vector<Satellite*> - the vector of Satellites to set to. |
|---|--|

Returns

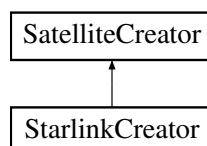
void

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[SimulationState.h](#)
- C:/Users/labuser2/Downloads/System/System/[SimulationState.cpp](#)

5.34 StarlinkCreator Class Reference

Inheritance diagram for StarlinkCreator:



Public Member Functions

- **StarlinkCreator ()**
Constructor for [StarlinkCreator](#) objects. Sets IDcount to zero.
- **Satellite * factoryMethod ()**
Factory method to create [StarlinkSatellite](#) objects.

- void [setIDCount](#) (int count)

Set the current [StarlinkSatellite](#) ID counter.

5.34.1 Member Function Documentation

5.34.1.1 factoryMethod()

```
Satellite * StarlinkCreator::factoryMethod ( ) [virtual]
```

Factory method to create [StarlinkSatellite](#) objects.

Returns

Satellite*

Implements [SatelliteCreator](#).

5.34.1.2 setIDCount()

```
void StarlinkCreator::setIDCount (
    int count ) [virtual]
```

Set the current [StarlinkSatellite](#) ID counter.

Parameters

| | |
|--------------|--------------------------------|
| <i>count</i> | int - the number to be set to. |
|--------------|--------------------------------|

Returns

void

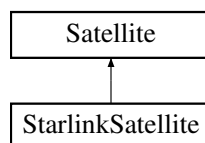
Implements [SatelliteCreator](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[StarlinkCreator.h](#)
- C:/Users/labuser2/Downloads/System/System/[StarlinkCreator.cpp](#)

5.35 StarlinkSatellite Class Reference

Inheritance diagram for StarlinkSatellite:



Public Member Functions

- [StarlinkSatellite](#) (int ID)
Constructor method for [StarlinkSatellite](#) objects. Sets the ID to the integer sent in as a parameter.
- [StarlinkSatellite](#) ([Satellite](#) *s, int id)
Copy constructor for [StarlinkSatellite](#) objects.
- [SatelliteState](#) * [getState](#) ()
Returns the state of the [StarlinkSatellite](#).
- void [setState](#) ([SatelliteState](#) *s)

Sets the state of the [Satellite](#) to the [SatelliteState](#) sent in as a parameter.

- [Satellite](#) * clone (int id)

Calls the copy constructor in order to replicate current satellite.

Additional Inherited Members

5.35.1 Constructor & Destructor Documentation

5.35.1.1 StarlinkSatellite() [1/2]

```
StarlinkSatellite::StarlinkSatellite (
    int ID )
```

Constructor method for [StarlinkSatellite](#) objects. Sets the ID to the integer sent in as a parameter.

Parameters

| | |
|-----------|-----|
| <i>ID</i> | int |
|-----------|-----|

5.35.1.2 StarlinkSatellite() [2/2]

```
StarlinkSatellite::StarlinkSatellite (
    Satellite * s,
    int id )
```

Copy constructor for [StarlinkSatellite](#) objects.

Parameters

| | |
|-----------|-------------------------------------|
| <i>s</i> | Satellite* - the satellite to copy. |
| <i>id</i> | int - the id of the new satellite. |

5.35.2 Member Function Documentation

5.35.2.1 clone()

```
Satellite * StarlinkSatellite::clone (
    int id ) [virtual]
```

Calls the copy constructor in order to replicate current satellite.

Parameters

| | |
|-----------|------------------------------------|
| <i>id</i> | int - the id of the new satellite. |
|-----------|------------------------------------|

Returns

Satellite*

Reimplemented from [Satellite](#).

5.35.2.2 getState()

```
SatelliteState * StarlinkSatellite::getState ( ) [virtual]
```

Returns the state of the [StarlinkSatellite](#).

Returns

SatelliteState*

Reimplemented from [Satellite](#).

5.35.2.3 setState()

```
void StarlinkSatellite::setState (
    SatelliteState * s )
```

Sets the state of the [Satellite](#) to the [SatelliteState](#) sent in as a parameter.

Parameters

| | |
|---|--------------------------------|
| s | SatelliteState |
|---|--------------------------------|

Returns

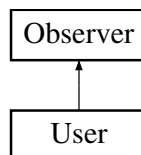
void

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[StarlinkSatellite.h](#)
- C:/Users/labuser2/Downloads/System/System/[StarlinkSatellite.cpp](#)

5.36 User Class Reference

Inheritance diagram for User:



Public Member Functions

- [User](#) ([Satellite](#) *s)
Constructor for [User](#) objects. Sets the subject variable to the [StarlinkSatellite](#) passed in as a parameter.
- void [update](#) (int satellitelID, string status)
Updates the satelliteState variable.

5.36.1 Constructor & Destructor Documentation

5.36.1.1 User()

```
User::User (
    Satellite * s )
```

Constructor for [User](#) objects. Sets the subject variable to the [StarlinkSatellite](#) passed in as a parameter.

Parameters

| | |
|---|---|
| s | StarlinkSatellite* - the StarlinkSatellite to set to. |
|---|---|

5.36.2 Member Function Documentation

5.36.2.1 update()

```
void User::update (
    int satelliteID,
    string status ) [virtual]
```

Updates the satelliteState variable.

Returns

void

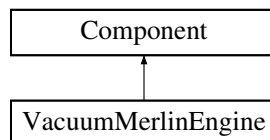
Implements [Observer](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[User.h](#)
- C:/Users/labuser2/Downloads/System/System/[User.cpp](#)

5.37 VacuumMerlinEngine Class Reference

Inheritance diagram for VacuumMerlinEngine:



Public Member Functions

- **VacuumMerlinEngine ()**
Constructor for [VacuumMerlinEngine](#) objects. Calls the [Component](#) constructor to initialize the attributes.
- void [simulate](#) ()
Starts the simulation for [VacuumMerlinEngine](#) objects.
- void [test](#) ()
Tests if the [MerlinEngine](#) meets all the requirements for a successful launch. The requirements:
- void [fireVacuumMerlin](#) ()
Outputs that a VacuumMerlin object has been ignited.

Additional Inherited Members

5.37.1 Member Function Documentation

5.37.1.1 fireVacuumMerlin()

```
void VacuumMerlinEngine::fireVacuumMerlin ( ) [virtual]
```

Outputs that a VacuumMerlin object has been ignited.

Returns

void

Reimplemented from [Component](#).

5.37.1.2 simulate()

```
void VacuumMerlinEngine::simulate ( ) [virtual]
```

Starts the simulation for [VacuumMerlinEngine](#) objects.

Returns

void

Reimplemented from [Component](#).

5.37.1.3 test()

```
void VacuumMerlinEngine::test ( ) [virtual]
```

Tests if the [MerlinEngine](#) meets all the requirements for a successful launch. The requirements:

- the cost must be >0
- it must have a capsuleType
- it must have a rocketType

Returns

void

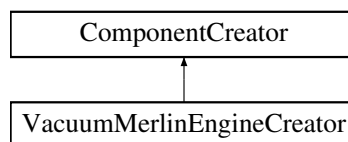
Reimplemented from [Component](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[VacuumMerlinEngine.h](#)
- C:/Users/labuser2/Downloads/System/System/[VacuumMerlinEngine.cpp](#)

5.38 VacuumMerlinEngineCreator Class Reference

Inheritance diagram for VacuumMerlinEngineCreator:



Public Member Functions

- [Component](#) * [factoryMethod](#) ()
Factory Method to create [VacuumMerlinEngine](#) objects.

5.38.1 Member Function Documentation

5.38.1.1 factoryMethod()

```
Component * VacuumMerlinEngineCreator::factoryMethod ( ) [virtual]
```

Factory Method to create [VacuumMerlinEngine](#) objects.

Returns

[Component](#)*

Implements [ComponentCreator](#).

The documentation for this class was generated from the following files:

- C:/Users/labuser2/Downloads/System/System/[VacuumMerlinEngineCreator.h](#)
- C:/Users/labuser2/Downloads/System/System/[VacuumMerlinEngineCreator.cpp](#)

Chapter 6

File Documentation

6.1 C:/Users/labuser2/Downloads/System/System/Broadcasting.cpp File Reference

Implementation for [Broadcasting.h](#).

```
#include "Broadcasting.h"
#include "Offline.h"
```

6.1.1 Detailed Description

Implementation for [Broadcasting.h](#).

6.2 C:/Users/labuser2/Downloads/System/System/Broadcasting.h File Reference

Participant - Concrete State (State). Describes the properties and methods of a [Satellite](#) in the 'broadcasting' state.

```
#include "SatelliteState.h"
```

Classes

- class [Broadcasting](#)

6.2.1 Detailed Description

Participant - Concrete State (State). Describes the properties and methods of a [Satellite](#) in the 'broadcasting' state.

Author

The 6 Musketeers

6.3 Broadcasting.h

[Go to the documentation of this file.](#)

```
1
10 #ifndef BROADCASTING_H
11 #define BROADCASTING_H
12
13 #include "SatelliteState.h"
14
15 using namespace std;
16 class Satellite;
17 class Broadcasting : public SatelliteState
18 {
19     public:
23     Broadcasting();
```

```

24
29     string getType();
30
35     SatelliteState* handleChange();
36 };
37
38 #endif

```

6.4 C:/Users/labuser2/Downloads/System/System/Builder.h File Reference

Participant - [Builder](#) ([Builder](#)) Describes the methods to build the components of a rocket.

```

#include <iostream>
#include <string>
#include "Component.h"
#include "Simulation.h"

```

Classes

- class [Builder](#)

6.4.1 Detailed Description

Participant - [Builder](#) ([Builder](#)) Describes the methods to build the components of a rocket.

Author

The 6 Musketeers

6.5 Builder.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef BUILDER_H
10 #define BUILDER_H
11
12 #include <iostream>
13 #include <string>
14 #include "Component.h"
15 #include "Simulation.h"
16
17 using namespace std;
18 class Builder
19 {
20     public:
26     virtual void buildFalcon9() = 0;
27
33     virtual void buildFalconHeavy() = 0;
34
40     virtual void constructCapsule(string c) = 0;
41
42
48     virtual Component* getSpacecraft() = 0;
49
55     virtual Simulation* createSimulation() = 0;
56 };
57
58 #endif

```

6.6 C:/Users/labuser2/Downloads/System/System/CapsuleArriving.cpp File Reference

Implementation for [CapsuleArriving.h](#).

```

#include "CapsuleArriving.h"
#include "CapsuleDocked.h"

```


6.6.1 Detailed Description

Implementation for [CapsuleArriving.h](#).

6.7 C:/Users/labuser2/Downloads/System/System/CapsuleArriving.h File Reference

Participant - Concrete State (State) Describes the methods of a Capusle that is in an 'arriving' state.

```
#include "CapsuleState.h"
```

Classes

- class [CapsuleArriving](#)

6.7.1 Detailed Description

Participant - Concrete State (State) Describes the methods of a Capusle that is in an 'arriving' state.

Author

The 6 Musekteers

6.8 CapsuleArriving.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef CAPSULEARRIVING_H
10 #define CAPSULEARRIVING_H
11
12 #include "CapsuleState.h"
13
14 using namespace std;
15
16 class CapsuleArriving : public CapsuleState
17 {
18     public:
22     CapsuleArriving();
23
28     string getState();
29
34     CapsuleState* handleChange();
35 };
36
37 #endif
```

6.9 C:/Users/labuser2/Downloads/System/System/CapsuleDeparting.cpp File Reference

Implementation for [CapsuleDeparting.h](#).

```
#include "CapsuleDeparting.h"
#include "CapsuleArriving.h"
#include "RocketCapsule.h"
```

6.9.1 Detailed Description

Implementation for [CapsuleDeparting.h](#).

6.10 C:/Users/labuser2/Downloads/System/System/CapsuleDeparting.h File Reference

Participant - Concrete State (State) Describes the methods of a Capusle that is in a 'departing' state.
`#include "CapsuleState.h"`

Classes

- class [CapsuleDeparting](#)

6.10.1 Detailed Description

Participant - Concrete State (State) Describes the methods of a Capusle that is in a 'departing' state.

Author

The 6 Musekteers

6.11 CapsuleDeparting.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef CAPSULEDEPARTING_H
10 #define CAPSULEDEPARTING_H
11
12 #include "CapsuleState.h"
13
14 using namespace std;
15
16 class CapsuleDeparting : public CapsuleState
17 {
18     public:
19
23         CapsuleDeparting();
24
29         string getState();
30
35         CapsuleState* handleChange();
36 };
37
38 #endif
```

6.12 C:/Users/labuser2/Downloads/System/System/CapsuleDocked.cpp File Reference

Implementation for [CapsuleDocked.h](#).

```
#include "CapsuleDocked.h"
#include "CapsuleOffline.h"
#include "CapsuleState.h"
#include "RocketCapsule.h"
```

6.12.1 Detailed Description

Implementation for [CapsuleDocked.h](#).

6.13 C:/Users/labuser2/Downloads/System/System/CapsuleDocked.h File Reference

Participant - Concrete State (State) Describes the methods of a Capusle that is in a 'docked' state.
`#include "CapsuleState.h"`

Classes

- class [CapsuleDocked](#)

6.13.1 Detailed Description

Participant - Concrete State (State) Describes the methods of a Capusle that is in a 'docked' state.

Author

The 6 Musekteers

6.14 CapsuleDocked.h

[Go to the documentation of this file.](#)

```
1
2
3
4
5
6
7
8 #ifndef CAPSULEDOCKED_H
9 #define CAPSULEDOCKED_H
10
11 #include "CapsuleState.h"
12
13 using namespace std;
14
15 class CapsuleDocked : public CapsuleState
16 {
17     public:
18
19
20
21
22     CapsuleDocked();
23
24
25
26
27
28     string getState();
29
30
31
32
33
34     CapsuleState* handleChange();
35 };
36
37 #endif
```

6.15 C:/Users/labuser2/Downloads/System/System/CapsuleOffline.cpp File Reference

Implementation for [CapsuleOffline.h](#).
`#include "CapsuleOffline.h"`
`#include "CapsuleDeparting.h"`

6.15.1 Detailed Description

Implementation for [CapsuleOffline.h](#).

6.16 C:/Users/labuser2/Downloads/System/System/CapsuleOffline.h File Reference

Participant - Concrete State (State) Describes the methods of a Capusle that is in a 'docked' state.

```
#include "CapsuleState.h"
```

Classes

- class [CapsuleOffline](#)

6.16.1 Detailed Description

Participant - Concrete State (State) Describes the methods of a Capsule that is in a 'docked' state.

Author

The 6 Musekteers

6.17 CapsuleOffline.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef CAPSULEOFFLINE_H
10 #define CAPSULEOFFLINE_H
11
12 #include "CapsuleState.h"
13
14 using namespace std;
15
16 class CapsuleOffline : public CapsuleState
17 {
18     public:
22         CapsuleOffline();
23
28         string getState();
29
34         CapsuleState* handleChange();
35 };
36
37 #endif
```

6.18 C:/Users/labuser2/Downloads/System/System/CapsuleState.h File Reference

Participant - State (State) Describes the interface for the different states of a capsule.

```
#include "RocketCapsule.h"
```

Classes

- class [CapsuleState](#)

6.18.1 Detailed Description

Participant - State (State) Describes the interface for the different states of a capsule.

Author

The 6 Musekteers

6.19 CapsuleState.h

[Go to the documentation of this file.](#)

```
1
```

```
10 #ifndef CAPSULESTATE_H
11 #define CAPSULESTATE_H
12
13 #include "RocketCapsule.h"
14
15 using namespace std;
16 class CapsuleState
17 {
18     public:
24     virtual string getState() = 0;
25
31     virtual CapsuleState* handleChange() = 0;
32 };
33
34
35
36 #endif
```

6.20 C:/Users/labuser2/Downloads/System/System/Caretaker.cpp File Reference

Implementation for [Caretaker.h](#).
#include "Caretaker.h"

6.20.1 Detailed Description

Implementation for [Caretaker.h](#).

6.21 C:/Users/labuser2/Downloads/System/System/Caretaker.h File Reference

Participant - [Caretaker](#) ([Memento](#)) Describes the class responsible for the safekeeping of the [Memento](#) class' state.
#include <string>
#include <iostream>
#include <vector>
#include "Memento.h"

Classes

- class [Caretaker](#)

6.21.1 Detailed Description

Participant - [Caretaker](#) ([Memento](#)) Describes the class responsible for the safekeeping of the [Memento](#) class' state.

Author

The 6 Musketeers

6.22 Caretaker.h

[Go to the documentation of this file.](#)

```
1
11 #ifndef CARETAKER_H
12 #define CARETAKER_H
13
14 #include <string>
15 #include <iostream>
16 #include <vector>
17
```

```

18 #include "Memento.h"
19
20 using namespace std;
21
22 class Caretaker {
23
24 private:
25     vector<Memento> store;
27 public:
28
29     Caretaker();
30
31     ~Caretaker();
32
33     void storeMemento(Memento* m);
34
35     Memento* retrieveMemento();
36
37     int getSize();
38 };
39
40 #endif

```

6.23 C:/Users/labuser2/Downloads/System/System/CargoDragon.cpp File Reference

Implementation for [CargoDragon.h](#).

```
#include "CargoDragon.h"
```

6.23.1 Detailed Description

Implementation for [CargoDragon.h](#).

6.24 C:/Users/labuser2/Downloads/System/System/CargoDragon.h File Reference

Participant - Concrete Decorator (Decorator) Defines the attributes and methods for a [RocketCapsule](#) that carries Cargo.

```
#include "RocketCapsule.h"
```

Classes

- class [CargoDragon](#)

6.24.1 Detailed Description

Participant - Concrete Decorator (Decorator) Defines the attributes and methods for a [RocketCapsule](#) that carries Cargo.

Author

The 6 Musketeers

6.25 CargoDragon.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef CARGODRAGON_H
10 #define CARGODRAGON_H
11
12 #include "RocketCapsule.h"
13
14 class CargoDragon : public RocketCapsule
15 {

```

```

16     public:
22         CargoDragon(Component* r);
23
28         void simulate();
29
38         void test();
39 };
40
41 #endif

```

6.26 C:/Users/labuser2/Downloads/System/System/CommNetwork.cpp File Reference

Implementation for [CommNetwork.h](#).

```

#include "CommNetwork.h"
#include <iterator>

```

6.26.1 Detailed Description

Implementation for [CommNetwork.h](#).

6.27 C:/Users/labuser2/Downloads/System/System/CommNetwork.h File Reference

Participant - Concrete [Mediator](#) ([Mediator](#)) Defines the attributes and methods for the class used for communication between the satellites.

```

#include <string>
#include <vector>
#include "Mediator.h"
#include "Satellite.h"

```

Classes

- class [CommNetwork](#)

6.27.1 Detailed Description

Participant - Concrete [Mediator](#) ([Mediator](#)) Defines the attributes and methods for the class used for communication between the satellites.

Author

The 6 Musketeers

6.28 CommNetwork.h

[Go to the documentation of this file.](#)

```

1
10 #ifndef COMMNETWORK_H
11 #define COMMNETWORK_H
12
13 #include <string>
14 #include <vector>
15 #include "Mediator.h"
16 #include "Satellite.h"
17
18 class CommNetwork : public Mediator {
19     public:
20         vector<Satellite*> colleagueList;
22     public:
23         CommNetwork(vector<Satellite*>);
29         void notify(int sender);

```

```

30
38         void sendMessage(int sender, int receiver, string msg);
39     };
40
41 #endif

```

6.29 C:/Users/labuser2/Downloads/System/System/Component.cpp File Reference

Implementation for [Component.h](#).

```

#include "Component.h"
#include "Facade.h"

```

6.29.1 Detailed Description

Implementation for [Component.h](#).

6.30 C:/Users/labuser2/Downloads/System/System/Component.h File Reference

Participant - [Component](#) (Decorator), [Component](#) (Composite), Client (Chain of Responsibility), Product ([Builder](#)), Product (Factory Method), Client (Prototype), Implementor (Brdige) Defines the attributes and methods for [Component](#) objects.

```

#include <string>
#include <iostream>

```

Classes

- class [Component](#)

6.30.1 Detailed Description

Participant - [Component](#) (Decorator), [Component](#) (Composite), Client (Chain of Responsibility), Product ([Builder](#)), Product (Factory Method), Client (Prototype), Implementor (Brdige) Defines the attributes and methods for [Component](#) objects.

Author

The 6 Musketeers

6.31 Component.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef COMPONENT_H
10 #define COMPONENT_H
11
12 #include <string>
13 #include <iostream>
14
15 using namespace std;
16
17 class Component
18 {
19     protected:
20         double cost;
21         //string rocketType;    /**< The type of the rocket */
22         //string capsuleType;  /**< The type of the capsule */
23
24     public:
25         Component(double c);
26
27
28
29
30

```



```

36     virtual void simulate();
37
43     virtual void test();
44
50     virtual void add(Component* c);
51
57     virtual void remove(int pos);
58
64     virtual Component* getComponent(int pos);
65
66
67
68     //Component* clone();
69
70
71
72
77     double getCost();
78
83     virtual void separate();
84
90     virtual void fireMerlin();
91
96     virtual void land();
97
102     virtual int getSize();
103
108     virtual void fireVacuumMerlin();
109 };
110
111 static float rocketCost;
113 #endif

```

6.32 C:/Users/labuser2/Downloads/System/System/ComponentComposite.cpp File Reference↔

Implementation for [ComponentComposite.h](#).

```
#include "ComponentComposite.h"
```

6.32.1 Detailed Description

Implementation for [ComponentComposite.h](#).

6.33 C:/Users/labuser2/Downloads/System/System/ComponentComposite.h File Reference↔

Participant - Handler (Chain of Responsibility) Defines an interface to handle the requests and implementatins of the Falcon9 and FalconHeavy components.

```
#include "Component.h"
```

```
#include <vector>
```

Classes

- class [ComponentComposite](#)

6.33.1 Detailed Description

Participant - Handler (Chain of Responsibility) Defines an interface to handle the requests and implementatins of the Falcon9 and FalconHeavy components.

Author

The 6 Muskateers

6.34 ComponentComposite.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef COMPONENTCOMPOSITE_H
10 #define COMPONENTCOMPOSITE_H
11
12 #include "Component.h"
13 #include <vector>
14
15 class ComponentComposite : public Component
16 {
17     private:
18         vector<Component*> components;
19     public:
20         ComponentComposite();
21
22         void simulate();
23
24         void test();
25
26         virtual void add(Component* c);
27
28         virtual void remove(int pos);
29
30         Component* getComponent(int pos);
31
32         int getSize();
33
34         void separate();
35
36         void fireMerlin();
37
38         void land();
39 };
40
41 #endif

```

6.35 C:/Users/labuser2/Downloads/System/System/ComponentCreator.h

File Reference

Participant - Creator (Factory Method), Prototype (Prototype). Defines the interface creating the Merlin, Vacuum Merlin and Core Engines.

```
#include "Component.h"
```

Classes

- class [ComponentCreator](#)

6.35.1 Detailed Description

Participant - Creator (Factory Method), Prototype (Prototype). Defines the interface creating the Merlin, Vacuum Merlin and Core Engines.

Author

The 6 Musketeers

6.36 ComponentCreator.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef COMPONENTCREATOR_H
10 #define COMPONENTCREATOR_H
11
12 #include "Component.h"
13
14 class ComponentCreator
15 {
16     private:
17         Component* component;

```

```

19     public:
25         virtual Component* factoryMethod() = 0;
26
27
28
29         //virtual Component* clone(Component* C) = 0;
30 };
31
32 #endif

```

6.37 C:/Users/labuser2/Downloads/System/System/ConcreteRocketBuilder.cpp File Reference

Implementation for [ConcreteRocketBuilder.h](#).

```

#include "ConcreteRocketBuilder.h"
#include "CommNetwork.h"

```

6.37.1 Detailed Description

Implementation for [ConcreteRocketBuilder.h](#).

6.38 C:/Users/labuser2/Downloads/System/System/ConcreteRocketBuilder.h File Reference

Participant - Concrete [Builder \(Builder\)](#) Defines the methods and attributes of the class that builds a rocket.

```

#include "Component.h"
#include "Builder.h"
#include "ComponentCreator.h"
#include "CoreCreator.h"
#include "VacuumMerlinEngineCreator.h"
#include "MerlinEngineCreator.h"
#include "ComponentComposite.h"
#include "RocketCapsule.h"
#include "CrewDragon.h"
#include "CargoDragon.h"
#include "Fairing.h"
#include "SimulationState.h"
#include "SatelliteCreator.h"
#include "StarlinkCreator.h"
#include "User.h"
#include "Simulation.h"
#include <iostream>
#include <string>
#include <vector>
#include <cstdlib>

```

Classes

- class [ConcreteRocketBuilder](#)

6.38.1 Detailed Description

Participant - Concrete [Builder \(Builder\)](#) Defines the methods and attributes of the class that builds a rocket.

Author

The 6 Musketeers

6.39 ConcreteRocketBuilder.h[Go to the documentation of this file.](#)

```

1
9 #ifndef CONCRETEROCKETBUILDER_H
10 #define CONCRETEROCKETBUILDER_H
11
12 #include "Component.h"
13 #include "Builder.h"
14 #include "ComponentCreator.h"
15 #include "CoreCreator.h"
16 #include "VacuumMerlinEngineCreator.h"
17 #include "MerlinEngineCreator.h"
18 #include "ComponentComposite.h"
19 #include "RocketCapsule.h"
20 #include "CrewDragon.h"
21 #include "CargoDragon.h"
22 #include "Fairing.h"
23 #include "SimulationState.h"
24 #include "SatelliteCreator.h"
25 #include "StarlinkCreator.h"
26 #include "User.h"
27
28 #include "Simulation.h"
29
30 #include <iostream>
31 #include <string>
32 #include <vector>
33 #include <cstdlib>
34
35 class ConcreteRocketBuilder : public Builder
36 {
37     private:
38         ComponentCreator* merlinCreator;
39         ComponentCreator* vacuumMerlinCreator;
40         ComponentCreator* coreCreator;
41         SatelliteCreator* starlinkCreator;
42         Component* rocket;
43         RocketCapsule* capsule;
44         SimulationState* simulationState;
45     public:
46         ConcreteRocketBuilder();
47
48         Component* getSpacecraft();
49
50         RocketCapsule* getCapsule();
51
52         void buildFalcon9();
53
54         void buildFalconHeavy();
55
56         void constructCapsule(string type);
57
58         Simulation* createSimulation();
59 };
60
61 #endif

```

6.40 C:/Users/labuser2/Downloads/System/System/CoreCreator.cpp File Reference

Implementation for Cor.Creatorh.

#include "CoreCreator.h"

6.40.1 Detailed Description

Implementation for Cor.Creatorh.

6.41 C:/Users/labuser2/Downloads/System/System/CoreCreator.h File Reference

Participant - Concrete Creator (Factory Method), ConcretePrototype (Prototype). Defines the methods and attributes of the class that builds [FalconCore](#) engines.

```
#include "ComponentCreator.h"
#include "Component.h"
#include "FalconCore.h"
```

Classes

- class [CoreCreator](#)

6.41.1 Detailed Description

Participant - Concrete Creator (Factory Method), ConcretePrototype (Prototype). Defines the methods and attributes of the class that builds [FalconCore](#) engines.

Author

The 6 Musketeers

6.42 CoreCreator.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef CORECREATOR_H
10 #define CORECREATOR_H
11
12 #include "ComponentCreator.h"
13 #include "Component.h"
14 #include "FalconCore.h"
15
16 //Factory Method Concrete Creator
17 class CoreCreator : public ComponentCreator
18 {
19     public:
24         Component* factoryMethod();
25
26         //Component* clone(Component* C);
27 };
28
29 #endif
```

6.43 C:/Users/labuser2/Downloads/System/System/CrewDragon.cpp File Reference

Implementation for [CrewDragon.h](#).

```
#include "CrewDragon.h"
```

6.43.1 Detailed Description

Implementation for [CrewDragon.h](#).

6.44 C:/Users/labuser2/Downloads/System/System/CrewDragon.h File Reference

Participant - Concrete Decorator (Decorator) Defines the attributes and methods for a Capsule carrying crew members.

```
#include "RocketCapsule.h"
#include <vector>
```

Classes

- class [CrewDragon](#)

6.44.1 Detailed Description

Participant - Concrete Decorator (Decorator) Defines the attributes and methods for a Capsule carrying crew members.

Author

The 6 Musketeers

6.45 CrewDragon.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef CREWDRAGON_H
10 #define CREWDRAGON_H
11
12 #include "RocketCapsule.h"
13
14 #include <vector>
15
16 class CrewDragon : public RocketCapsule
17 {
18     private:
19         vector<string> passengers;
20     public:
21         CrewDragon(Component* r);
22
23         void simulate();
24
25         void test();
26
27         vector<string> getPassengers();
28
29         void setPassengers(vector<string> p);
30 };
31
32 #endif
```

6.46 C:/Users/labuser2/Downloads/System/System/Director.cpp File Reference

Implementation for [Director.h](#).

```
#include "Director.h"
```

6.46.1 Detailed Description

Implementation for [Director.h](#).

6.47 C:/Users/labuser2/Downloads/System/System/Director.h File Reference

Participant - [Director](#) ([Builder](#)) Defines the attributes and methods for the class that constructs rockets using the [Builder](#) interface.

```
#include <string>
#include <iostream>
```

```
#include "Builder.h"
```

Classes

- class [Director](#)

6.47.1 Detailed Description

Participant - [Director](#) ([Builder](#)) Defines the attributes and methods for the class that constructs rockets using the [Builder](#) interface.

Author

The 6 Musketeers

6.48 Director.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef DIRECTOR_H
10 #define DIRECTOR_H
11
12 #include <string>
13 #include <iostream>
14
15 #include "Builder.h"
16
17 using namespace std;
18
19 class Director
20 {
21     private:
22         string type;
23         Builder* builder;
24     public:
25         Director (Builder* b);
26
27         ~Director();
28
29         Component* construct();
30
31         void constructCapsule();
32
33         Simulation* createSimulation();
34 };
35 #endif //DIRECTOR_H
```

6.49 C:/Users/labuser2/Downloads/System/System/Facade.cpp File Reference

Implementation for [Facade.h](#).

```
#include "Facade.h"
#include "ConcreteRocketBuilder.h"
#include "Director.h"
#include "Caretaker.h"
```

6.49.1 Detailed Description

Implementation for [Facade.h](#).

6.50 C:/Users/labuser2/Downloads/System/System/Facade.h File Reference

Participant - [Facade](#) ([Facade](#)) Delegates client requests to appropriate subsystem objects.

```
#include "Simulation.h"
#include "Online.h"
#include "Offline.h"
#include "Broadcasting.h"
#include "RocketCapsule.h"
```

Classes

- class [Facade](#)

6.50.1 Detailed Description

Participant - [Facade](#) ([Facade](#)) Delegates client requests to appropriate subsystem objects.

Author

The 6 Musketeers

6.51 Facade.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef FACADE_H
10 #define FACADE_H
11
12 #include "Simulation.h"
13 #include "Online.h"
14 #include "Offline.h"
15 #include "Broadcasting.h"
16 #include "RocketCapsule.h"
17
18 class Facade
19 {
20     private:
21         Simulation* simulation;
22         Component* rocket;
23     public:
24         Facade();
25
26         ~Facade();
27
28         //Bridge (Simulation) subsystem
29
30         void launch();
31
32         void test();
33
34         //Builder subsystem
35         void build();
36
37         //Memento subsystem
38
39         void storeSimulation();
40
41         void retrieveSimulation();
42
43         void useCommNetwork();
44
45         void separateBoosters();
46
47         Component* getRocket();
48
49         void fireMerlin();
50
51         void fireVacuumMerlin();
52
53         void runSimulation();
54
55         void deliverCrew();
56
57         void distributeSatellites();
58
59         void staticFireTest();
60
61         void jettisonFairing();
62
63 }
```



```

139         void printSimulation();
140
145         bool editSimulation();
146
151         void retrieveAll();
152     };
153
154 #endif

```

6.52 C:/Users/labuser2/Downloads/System/System/Fairing.cpp File Reference

Implementation for [Fairing.h](#).

```
#include "Fairing.h"
```

6.52.1 Detailed Description

Implementation for [Fairing.h](#).

6.53 C:/Users/labuser2/Downloads/System/System/Fairing.h File Reference

Participant - Concrete Decorator (Decorator) Defines the attributes and methods for a [RocketCapsule](#) of type [Fairing](#).

```
#include "RocketCapsule.h"
```

```
#include "Satellite.h"
```

```
#include <vector>
```

Classes

- class [Fairing](#)

6.53.1 Detailed Description

Participant - Concrete Decorator (Decorator) Defines the attributes and methods for a [RocketCapsule](#) of type [Fairing](#).

Author

The 6 Musketeers

6.54 Fairing.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef FAIRING_H
10 #define FAIRING_H
11
12 #include "RocketCapsule.h"
13 #include "Satellite.h"
14
15 #include <vector>
16
17 class Fairing : public RocketCapsule
18 {
19     private:
20         vector<Satellite*> satellites;
22     public:
27         Fairing(Component* r);
28
33         void simulate();
42         void test();
43
48         vector<Satellite*> getSatellites();
49
54         void setSatellites(vector<Satellite*> s);

```

```
55
61     Satellite* getSatellite(int id);
62 };
63
64 #endif
```

6.55 C:/Users/labuser2/Downloads/System/System/FalconCore.cpp File Reference

Implementation for [FalconCore.h](#).

```
#include "FalconCore.h"
```

6.55.1 Detailed Description

Implementation for [FalconCore.h](#).

6.56 C:/Users/labuser2/Downloads/System/System/FalconCore.h File Reference

Participant - ConcreteComponent (Decorator), Leaf (Composite), ConcreteProduct (Factory Method), Concrete Implementor (Bridge) Defines the methods of the class that defines a [FalconCore](#) engine.

```
#include "Component.h"
```

Classes

- class [FalconCore](#)

6.56.1 Detailed Description

Participant - ConcreteComponent (Decorator), Leaf (Composite), ConcreteProduct (Factory Method), Concrete Implementor (Bridge) Defines the methods of the class that defines a [FalconCore](#) engine.

Author

The 6 Musketeers

6.57 FalconCore.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef FALCONCORE_H
10 #define FALCONCORE_H
11
12 #include "Component.h"
13
14 class FalconCore : public Component
15 {
16     public:
20         FalconCore();
21
26         void simulate();
27
36         void test();
37
42         void separate();
43
48         void land();
49 };
50
51 #endif
```

6.58 C:/Users/labuser2/Downloads/System/System/main.cpp File Reference

Runs the program.

```
#include "Facade.h"
#include <stdio.h>
#include <unistd.h>
```

Functions

- `int main (int argc, char **argv)`

6.58.1 Detailed Description

Runs the program.

6.59 mainpage.h

1

6.60 C:/Users/labuser2/Downloads/System/System/Mediator.h File Reference

Participant - [Mediator](#) ([Mediator](#)) Defines the methods of the interface that enables communication between the different satellites.

```
#include <string>
#include <iostream>
#include "Satellite.h"
```

Classes

- class [Mediator](#)

6.60.1 Detailed Description

Participant - [Mediator](#) ([Mediator](#)) Defines the methods of the interface that enables communication between the different satellites.

Author

The 6 Musketeers

6.61 Mediator.h

[Go to the documentation of this file.](#)

```
1
10 #ifndef MEDIATOR_H
11 #define MEDIATOR_H
12
13 #include <string>
14 #include <iostream>
15
16 #include "Satellite.h"
17 class Satellite;
18 using namespace std;
19
20 class Mediator {
21
22 private:
```

```

29 public:
36     virtual void notify(int sender) = 0;
37
45     virtual void sendMessage(int sender, int receiver, string msg) = 0;
46 };
47
48 #endif

```

6.62 C:/Users/labuser2/Downloads/System/System/Memento.cpp File Reference

Implementation for [Memento.h](#).

```
#include "Memento.h"
```

6.62.1 Detailed Description

Implementation for [Memento.h](#).

6.63 C:/Users/labuser2/Downloads/System/System/Memento.h File Reference

Participant - [Memento](#) ([Memento](#)) Defines the methods of the class that stores the state of the simulation of a rocket.

```
#include <iostream>
```

```
#include <string>
```

```
#include "SimulationState.h"
```

Classes

- class [Memento](#)

6.63.1 Detailed Description

Participant - [Memento](#) ([Memento](#)) Defines the methods of the class that stores the state of the simulation of a rocket.

Author

The 6 Musketeers

6.64 Memento.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef MEMENTO_H
10 #define MEMENTO_H
11
12 #include <iostream>
13 #include <string>
14
15 #include "SimulationState.h"
16
17 using namespace std;
18
19 class Memento
20 {
21     private:
22         SimulationState* state;
23     public:
24         SimulationState* getState();
25
26         void setState(SimulationState* c);
27
28         ~Memento();
29 };
30
31 #endif

```

6.65 C:/Users/labuser2/Downloads/System/System/MerlinEngine.cpp File Reference

Implementation for [MerlinEngine.h](#).

```
#include "MerlinEngine.h"
```

6.65.1 Detailed Description

Implementation for [MerlinEngine.h](#).

6.66 C:/Users/labuser2/Downloads/System/System/MerlinEngine.h File Reference

Participant - ConcreteProduct (Factory Method), Concrete Implementor (Bridge). Defines the methods of the class that defines a Merlin engine.

```
#include "Component.h"
```

Classes

- class [MerlinEngine](#)

6.66.1 Detailed Description

Participant - ConcreteProduct (Factory Method), Concrete Implementor (Bridge). Defines the methods of the class that defines a Merlin engine.

Author

The 6 Musketeers

6.67 MerlinEngine.h

[Go to the documentation of this file.](#)

```
1
10 #ifndef MERLINENGINE_H
11 #define MERLINENGINE_H
12
13 #include "Component.h"
14
15 class MerlinEngine : public Component
16 {
17     public:
22         MerlinEngine();
23
28         void simulate();
29
38         void test();
39
44         void fireMerlin();
45 };
46
47 #endif
```

6.68 C:/Users/labuser2/Downloads/System/System/MerlinEngineCreator.cpp File Reference

Implementation for [MerlinEngineCreator.h](#).

```
#include "MerlinEngineCreator.h"
```

6.68.1 Detailed Description

Implementation for [MerlinEngineCreator.h](#).

6.69 C:/Users/labuser2/Downloads/System/System/MerlinEngineCreator.h File Reference

Participant - ConcreteCreator (Factory Method), ConcretePrototype (Prototype) Defines the methods of the class that creates [MerlinEngine](#) objects.

```
#include "ComponentCreator.h"
#include "Component.h"
#include "MerlinEngine.h"
```

Classes

- class [MerlinEngineCreator](#)

6.69.1 Detailed Description

Participant - ConcreteCreator (Factory Method), ConcretePrototype (Prototype) Defines the methods of the class that creates [MerlinEngine](#) objects.

Author

The 6 Musketeers

6.70 MerlinEngineCreator.h

[Go to the documentation of this file.](#)

```
1
10 #ifndef MERLINENGINECREATOR_H
11 #define MERLINENGINECREATOR_H
12
13 #include "ComponentCreator.h"
14 #include "Component.h"
15 #include "MerlinEngine.h"
16
17 class MerlinEngineCreator : public ComponentCreator
18 {
19     public:
24         Component* factoryMethod();
25
26
27         //Component* clone(Component* C);
28 };
29
30 #endif
```

6.71 C:/Users/labuser2/Downloads/System/System/Observer.h File Reference

Participant - [Observer](#) ([Observer](#)) Defines the methods of the abstract class that observes the state of a satellite.

```
#include <iostream>
#include <string>
```

Classes

- class [Observer](#)

6.71.1 Detailed Description

Participant - [Observer](#) ([Observer](#)) Defines the methods of the abstract class that observes the state of a satellite.

Author

The 6 Musketeers

6.72 Observer.h

[Go to the documentation of this file.](#)

```
1
10 #ifndef OBSERVER_H
11 #define OBSERVER_H
12
13 #include <iostream>
14 #include <string>
15
16 using namespace std;
17
18 class Observer {
19
20 public:
26     virtual void update(int satelliteID, string status) = 0;
27 };
28
29 #endif
```

6.73 C:/Users/labuser2/Downloads/System/System/Offline.cpp File Reference

Implementation for [Offline.h](#).

```
#include "Offline.h"
#include "Online.h"
```

6.73.1 Detailed Description

Implementation for [Offline.h](#).

6.74 C:/Users/labuser2/Downloads/System/System/Offline.h File Reference

Participant - Concrete State (State). Describes the properties and methods of a [Satellite](#) in the 'Offline' state.

```
#include "SatelliteState.h"
```

Classes

- class [Offline](#)

6.74.1 Detailed Description

Participant - Concrete State (State). Describes the properties and methods of a [Satellite](#) in the 'Offline' state.

Author

The 6 Musketeers

6.75 Offline.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef OFFLINE_H
10 #define OFFLINE_H
11
12 #include "SatelliteState.h"
13
14 using namespace std;
15
16 class Offline: public SatelliteState
17 {
18     public:
22         Offline();
23
28         string getType();
29
30
35         SatelliteState* handleChange();
36 };
37
38 #endif

```

6.76 C:/Users/labuser2/Downloads/System/System/Online.cpp File Reference

Implementation for [Online.h](#).

```

#include "Online.h"
#include "Broadcasting.h"

```

6.76.1 Detailed Description

Implementation for [Online.h](#).

6.77 C:/Users/labuser2/Downloads/System/System/Online.h File Reference

Participant - Concrete State (State). Describes the properties and methods of a [Satellite](#) in the 'Online' state.

```
#include "SatelliteState.h"
```

Classes

- class [Online](#)

6.77.1 Detailed Description

Participant - Concrete State (State). Describes the properties and methods of a [Satellite](#) in the 'Online' state.

Author

The 6 Musketeers

6.78 Online.h

[Go to the documentation of this file.](#)

```

1
10 #ifndef ONLINE_H
11 #define ONLINE_H
12
13 #include "SatelliteState.h"
14
15 using namespace std;

```



```

16
17 class Online : public SatelliteState
18 {
19     public:
20
24         Online();
25
30         string getType();
31
36         SatelliteState* handleChange();
37 };
38
39 #endif

```

6.79 C:/Users/labuser2/Downloads/System/System/RocketCapsule.cpp File Reference

Implementation for [RocketCapsule.h](#).

```

#include "RocketCapsule.h"
#include "CapsuleOffline.h"

```

6.79.1 Detailed Description

Implementation for [RocketCapsule.h](#).

6.80 C:/Users/labuser2/Downloads/System/System/RocketCapsule.h File Reference

Participant - Decorator (Decorator), Context (State) Describes the properties and methods of a RocketCapule that can be added to a rocket [Component](#).

```

#include <vector>
#include "Component.h"
#include "CapsuleState.h"
#include "Satellite.h"

```

Classes

- class [RocketCapsule](#)

6.80.1 Detailed Description

Participant - Decorator (Decorator), Context (State) Describes the properties and methods of a RocketCapule that can be added to a rocket [Component](#).

Author

The 6 Musketeers

6.81 RocketCapsule.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef ROCKETCAPSULE_H
10 #define ROCKETCAPSULE_H
11
12 #include <vector>
13
14 #include "Component.h"
15 #include "CapsuleState.h"
16 #include "Satellite.h"
17
18 class CapsuleState;

```

```

19
20 class RocketCapsule : public Component
21 {
22     protected:
23         string capsuleType;
24         CapsuleState* state;
25     private:
26         Component* rocket;
27         double payloadWeight;
28     public:
29         RocketCapsule(Component* r);
30
31         virtual void simulate() = 0;
32
33         virtual void test() = 0;
34
35         void addCapsule(Component* r);
36
37         void requestStateChange();
38
39         void setState(CapsuleState* s);
40
41         double getPayloadWeight();
42
43         void setPayloadWeight(double pw);
44
45         virtual void setPassengers(vector<string> p){};
46
47         virtual void setSatellites(vector<Satellite*> s){};
48
49         virtual Satellite* getSatellite(int id);
50
51         CapsuleState* getState();
52 };
53 #endif

```

6.82 C:/Users/labuser2/Downloads/System/System/Satellite.cpp File Reference

Implementation for [Satellite.h](#).

```
#include "Satellite.h"
```

6.82.1 Detailed Description

Implementation for [Satellite.h](#).

6.83 C:/Users/labuser2/Downloads/System/System/Satellite.h File Reference

Participant - ConcreteSubject ([Observer](#)), Colleague ([Mediator](#)), Context (State), Product (Factory Method). Describes the properties and methods of a [Satellite](#) object.

```

#include <iostream>
#include <vector>
#include <string>
#include "Mediator.h"
#include "Observer.h"
#include "Offline.h"

```

Classes

- class [Satellite](#)

6.83.1 Detailed Description

Participant - ConcreteSubject ([Observer](#)), Colleague ([Mediator](#)), Context (State), Product (Factory Method). Describes the properties and methods of a [Satellite](#) object.

Author

The 6 Muskateers

6.84 Satellite.h

[Go to the documentation of this file.](#)

```

1
10 #ifndef SATELLITE_H
11 #define SATELLITE_H
12
13 #include <iostream>
14 #include <vector>
15 #include <string>
16
17 #include "Mediator.h"
18 #include "Observer.h"
19 #include "Offline.h"
20
21 using namespace std;
22 class SatelliteState;
23 class Mediator;
24 class Satellite
25 {
26     protected:
27         SatelliteState* satelliteState;
28         Mediator* mediator;
29         vector<Observer*> observerList;
30         int ID;
31     public:
32         Satellite(int ID);
33
34         ~Satellite();
35
36         void changed();
37
38         //Mediator* getMediator();
39
40         int getID();
41
42         void sendMessage(int id, string msg);
43         void receiveMessage(int id, string msg);
44         void setMediator(Mediator* m);
45         Mediator* getMediator();
46         void requestStateChange();
47
48         void attach(Observer* o);
49         void detach(Observer* o);
50         void notify();
51
52         virtual SatelliteState* getState();
53         void setState(SatelliteState* s);
54         virtual Satellite* clone(int id){ return NULL; }
55 };
56 #endif

```

6.85 C:/Users/labuser2/Downloads/System/System/SatelliteCreator.cpp File Reference

Implementation for [SatelliteCreator.h](#).

```
#include "SatelliteCreator.h"
```

6.85.1 Detailed Description

Implementation for [SatelliteCreator.h](#).

6.86 C:/Users/labuser2/Downloads/System/System/SatelliteCreator.h File Reference

Participant - Creator (Factory Method). Describes the properties and methods the abstract class that creates a [Satellite](#).

```
#include <iostream>
#include <string>
#include "StarlinkSatellite.h"
```

Classes

- class [SatelliteCreator](#)

6.86.1 Detailed Description

Participant - Creator (Factory Method). Describes the properties and methods the abstract class that creates a [Satellite](#).

Author

The 6 Musketeers

6.87 SatelliteCreator.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef SATELLITECREATOR_H
10 #define SATELLITECREATOR_H
11
12 #include <iostream>
13 #include <string>
14
15 #include "StarlinkSatellite.h"
16
17 using namespace std;
18
19 class SatelliteCreator
20 {
21     private:
22         int count;
23     public:
24         SatelliteCreator();
25
26         virtual Satellite* factoryMethod()=0;
27
28         virtual void setIDCount(int id) = 0;
29 };
30
31 #endif
```

6.88 C:/Users/labuser2/Downloads/System/System/SatelliteState.h File Reference

Participant - State (State) Describes the interface for the different states of a [Satellite](#).

```
#include <string>
#include <iostream>
```

Classes

- class [SatelliteState](#)

6.88.1 Detailed Description

Participant - State (State) Describes the interface for the different states of a [Satellite](#).

Author

The 6 Musekteers

6.89 SatelliteState.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef SATELLITESTATE_H
10 #define SATELLITESTATE_H
11
12 #include <string>
13 #include <iostream>
14
15 using namespace std;
16
17 class SatelliteState
18 {
19     public:
25     virtual string getType() = 0;
26
32     virtual SatelliteState* handleChange() = 0;
33 };
34
35 #endif
```

6.90 C:/Users/labuser2/Downloads/System/System/Simulation.cpp File Reference

Implementation for [Simulation.h](#).

```
#include "Simulation.h"
#include "CapsuleDocked.h"
#include "User.h"
#include "Online.h"
#include "Offline.h"
#include "Broadcasting.h"
#include <stdio.h>
#include <unistd.h>
```

Functions

- void `animateRocket()`

Variables

- const char `rocket[]`

6.90.1 Detailed Description

Implementation for [Simulation.h](#).


```

15 #include "Memento.h"
16 #include "Satellite.h"
17 #include "SatelliteState.h"
18 #include "CapsuleArriving.h"
19 #include "CapsuleDeparting.h"
20 #include <cstring>
21 #include <vector>
22
23 class Simulation {
24
25 private:
26     SimulationState* simulationState;
27     Component* rocket;
28     RocketCapsule* capsule;
29     vector<string> methodCalls;
31 public:
32     Simulation(RocketCapsule* c, Component* r, SimulationState* s);
33
34     RocketCapsule* getCapsule();
35
36     Memento* createMemento();
37
38     void restoreMemento(Memento* m);
39
40     void staticFireTest();
41
42     void launch();
43
44     //virtual void tweakSimulation() = 0;
45
46     void printSimulation();
47
48     void jettisonFairing();
49
50     void separateBoosters();
51
52     void distributeSatellites();
53
54     void deliverCrew();
55
56     void sendMessage(int sender, int reciever, string message);
57
58     void runSimulation();
59
60     SimulationState* getState();
61
62     void fireMerlin();
63
64     void landBoosters();
65
66     void fireVacuumMerlin();
67
68     void changeSatelliteState(int id, SatelliteState* state);
69
70     void addCall(string c);
71
72     bool containsCall(string c);
73
74     void updateSimulationState();
75
76     void swapStage(int pos_1, int pos_2);
77
78     void removeStage(int pos);
79
80     int getSimulationSize();
81 };
82 #endif

```

6.93 C:/Users/labuser2/Downloads/System/System/SimulationState.cpp File Reference

Implementation for [SimulationState.h](#).

```
#include "SimulationState.h"
```

6.93.1 Detailed Description

Implementation for [SimulationState.h](#).

6.94 C:/Users/labuser2/Downloads/System/System/SimulationState.h File Reference

Participant - State ([Memento](#)) Describes the attributes and methods of a [SimulationState](#) object.

```
#include "Component.h"
#include "Satellite.h"
#include <string>
#include <iostream>
#include <vector>
```

Classes

- class [SimulationState](#)

6.94.1 Detailed Description

Participant - State ([Memento](#)) Describes the attributes and methods of a [SimulationState](#) object.

Author

The 6 Musketeers

6.95 SimulationState.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef SIMULATIONSTATE_H
10 #define SIMULATIONSTATE_H
11
12 #include "Component.h"
13 #include "Satellite.h"
14
15 #include <string>
16 #include <iostream>
17 #include <vector>
18
19 using namespace std;
20
21 class SimulationState {
22 private:
23     Component* rocket;
24     string capsuleType;
25     string rocketType;
26     double payloadWeight;
27     vector<Satellite*> satellites;
28     vector<string> passengers;
29     vector<string> methodCalls;
30 public:
31     SimulationState();
32
33     string getCapsuleType();
34
35     string getRocketType();
36
37     double getPayloadWeight();
38
39     vector<Satellite*> getSatellites();
40
41     vector<string> getPassengers();
42     vector<string> getMethodCalls();
43
44     void setCapsuleType(string s);
45
46     void setRocketType(string s);
47
48     void setPayloadWeight(double d);
49
50     void setSatellites(vector<Satellite*> s);
51
52     ~SimulationState() {
53         delete rocket;
54     }
55 }
```



```
106     void setPassengers(vector<string> p);
107
113     void setMethodCalls(vector<string> c);
114 };
115
116 #endif
```

6.96 C:/Users/labuser2/Downloads/System/System/StarlinkCreator.cpp File Reference

Implementation for [StarlinkCreator.h](#).

```
#include "StarlinkCreator.h"
```

6.96.1 Detailed Description

Implementation for [StarlinkCreator.h](#).

6.97 C:/Users/labuser2/Downloads/System/System/StarlinkCreator.h File Reference

Participant - Concrete Creator (Factory Method) Describes the attributes and methods of the class that creates [StarlinkSatellite](#) objects.

```
#include "SatelliteCreator.h"
```

```
#include "StarlinkSatellite.h"
```

Classes

- class [StarlinkCreator](#)

6.97.1 Detailed Description

Participant - Concrete Creator (Factory Method) Describes the attributes and methods of the class that creates [StarlinkSatellite](#) objects.

Author

The 6 Musketeers

6.98 StarlinkCreator.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef STARLINKCREATOR_H
10 #define STARLINKCREATOR_H
11
12 #include "SatelliteCreator.h"
13 #include "StarlinkSatellite.h"
14 class StarlinkCreator : public SatelliteCreator
15 {
16     private:
17         int IDcount;
19     public:
24         StarlinkCreator();
25
30         Satellite* factoryMethod();
31
37         void setIDCount(int count);
38 };
39
40 #endif
```

6.99 C:/Users/labuser2/Downloads/System/System/StarlinkSatellite.cpp File Reference

Implementation for [StarlinkSatellite.h](#).

```
#include "StarlinkSatellite.h"
```

6.99.1 Detailed Description

Implementation for [StarlinkSatellite.h](#).

6.100 C:/Users/labuser2/Downloads/System/System/StarlinkSatellite.h File Reference

Participant - Concrete Product (Factory Method) Describes the attributes and methods of [StarlinkSatellite](#) objects.

```
#include "Satellite.h"
```

```
#include "SatelliteState.h"
```

Classes

- class [StarlinkSatellite](#)

6.100.1 Detailed Description

Participant - Concrete Product (Factory Method) Describes the attributes and methods of [StarlinkSatellite](#) objects.

Author

The 6 Musketeers

6.101 StarlinkSatellite.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef STARLINKSATELLITE_H
10 #define STARLINKSATELLITE_H
11
12 #include "Satellite.h"
13 #include "SatelliteState.h"
14
15 class StarlinkSatellite : public Satellite
16 {
17     public:
23         StarlinkSatellite(int ID);
24
30         StarlinkSatellite(Satellite* s, int id);
31
36         SatelliteState* getState();
37
43         void setState(SatelliteState* s);
44
50         Satellite* clone(int id);
51 };
52
53 #endif
```

6.102 C:/Users/labuser2/Downloads/System/System/User.cpp File Reference

Implementation for [User.h](#).

```
#include "User.h"
```

6.102.1 Detailed Description

Implementation for [User.h](#).

6.103 C:/Users/labuser2/Downloads/System/System/User.h File Reference

Participant - Concrete [Observer](#) ([Observer](#)) Describes the attributes and methods of class that observes the state of [Satellite](#) objects.

```
#include "Observer.h"
#include "SatelliteState.h"
#include "StarlinkSatellite.h"
```

Classes

- class [User](#)

6.103.1 Detailed Description

Participant - Concrete [Observer](#) ([Observer](#)) Describes the attributes and methods of class that observes the state of [Satellite](#) objects.

Author

The 6 Musketeers

6.104 User.h

[Go to the documentation of this file.](#)

```
1
9 #ifndef USER_H
10 #define USER_H
11
12 #include "Observer.h"
13 #include "SatelliteState.h"
14 #include "StarlinkSatellite.h"
15
16 static int IDcounter = 0;
17
18 class User : public Observer
19 {
20     private:
21         SatelliteState* satelliteState;
22         Satellite* subject;
23         int ID;
24
25     public:
26         User(Satellite* s);
27
28         void update(int satelliteID, string status);
29 };
30
31 #endif
```

6.105 C:/Users/labuser2/Downloads/System/System/VacuumMerlin↵ Engine.cpp File Reference

Implementation for [VacuumMerlinEngine.h](#).

```
#include "VacuumMerlinEngine.h"
```

6.105.1 Detailed Description

Implementation for [VacuumMerlinEngine.h](#).

6.106 C:/Users/labuser2/Downloads/System/System/VacuumMerlin↔ Engine.h File Reference

Participant - Concrete Product (Factory Method) Describes the attributes and methods of a [VacuumMerlinEngine](#) object.

```
#include "Component.h"
```

Classes

- class [VacuumMerlinEngine](#)

6.106.1 Detailed Description

Participant - Concrete Product (Factory Method) Describes the attributes and methods of a [VacuumMerlinEngine](#) object.

Author

The 6 Musketeers

6.107 VacuumMerlinEngine.h

[Go to the documentation of this file.](#)

```
1
10 #ifndef VACUUMMERLINENGINE_H
11 #define VACUUMMERLINENGINE_H
12
13 #include "Component.h"
14
15 class VacuumMerlinEngine : public Component
16 {
17     public:
22     VacuumMerlinEngine();
23
28     void simulate();
29
38     void test();
39
44     void fireVacuumMerlin();
45 };
46
47 #endif
```

6.108 C:/Users/labuser2/Downloads/System/System/VacuumMerlin↔ EngineCreator.h File Reference

Participant - ConcretePrototype (Prototype), Concrete Implementor (Bridge), Concrete Product (Factory Method). Describes the attributes and methods of the class to create [VacuumMerlinEngine](#) objects.

```
#include "ComponentCreator.h"
#include "Component.h"
#include "VacuumMerlinEngine.h"
```

Classes

- class [VacuumMerlinEngineCreator](#)

6.108.1 Detailed Description

Participant - ConcretePrototype (Prototpe), Concrete Implementor (Bridge), Concrete Product (Factory Method). Describes the attributes and methods of the class to create [VacuumMerlinEngine](#) objects.

Author

The 6 Musketeers

6.109 VacuumMerlinEngineCreator.h

[Go to the documentation of this file.](#)

```
1
10 #ifndef VACUUMMERLINENGINECREATOR_H
11 #define VACUUMMERLINENGINECREATOR_H
12
13 #include "ComponentCreator.h"
14 #include "Component.h"
15 #include "VacuumMerlinEngine.h"
16
17 class VacuumMerlinEngineCreator : public ComponentCreator
18 {
19     public:
24         Component* factoryMethod();
25
26
27         //Component* clone(Component* C);
28 };
29
30 #endif
```


Index

add
 Component, [22](#)
 ComponentComposite, [25](#)
addCall
 Simulation, [59](#)
addCapsule
 RocketCapsule, [48](#)
attach
 Satellite, [53](#)

Broadcasting, [11](#)
 getType, [11](#)
 handleChange, [11](#)
build
 Facade, [34](#)
Builder, [12](#)
 buildFalcon9, [12](#)
 buildFalconHeavy, [12](#)
 constructCapsule, [12](#)
 createSimulation, [13](#)
 getSpacecraft, [13](#)
buildFalcon9
 Builder, [12](#)
 ConcreteRocketBuilder, [28](#)
buildFalconHeavy
 Builder, [12](#)
 ConcreteRocketBuilder, [28](#)

C:/Users/labuser2/Downloads/System/System/Broadcasting.cpp, [73](#)
C:/Users/labuser2/Downloads/System/System/Broadcasting.h, [73](#)
C:/Users/labuser2/Downloads/System/System/Builder.h, [74](#)
C:/Users/labuser2/Downloads/System/System/CapsuleArriving.cpp, [74](#)
C:/Users/labuser2/Downloads/System/System/CapsuleArriving.h, [75](#)
C:/Users/labuser2/Downloads/System/System/CapsuleDeparting.cpp, [75](#)
C:/Users/labuser2/Downloads/System/System/CapsuleDeparting.h, [76](#)
C:/Users/labuser2/Downloads/System/System/CapsuleDocked.cpp, [76](#)
C:/Users/labuser2/Downloads/System/System/CapsuleDocked.h, [77](#)
C:/Users/labuser2/Downloads/System/System/CapsuleOffline.cpp, [77](#)
C:/Users/labuser2/Downloads/System/System/CapsuleOffline.h, [77, 78](#)
C:/Users/labuser2/Downloads/System/System/CapsuleState.h, [78](#)
C:/Users/labuser2/Downloads/System/System/Caretaker.cpp, [79](#)
C:/Users/labuser2/Downloads/System/System/Caretaker.h, [79](#)
C:/Users/labuser2/Downloads/System/System/CargoDragon.cpp, [80](#)
C:/Users/labuser2/Downloads/System/System/CargoDragon.h, [80](#)
C:/Users/labuser2/Downloads/System/System/CommNetwork.cpp, [81](#)
C:/Users/labuser2/Downloads/System/System/CommNetwork.h, [81](#)
C:/Users/labuser2/Downloads/System/System/Component.cpp, [82](#)
C:/Users/labuser2/Downloads/System/System/Component.h, [82](#)
C:/Users/labuser2/Downloads/System/System/ComponentComposite.cpp, [83](#)
C:/Users/labuser2/Downloads/System/System/ComponentComposite.h, [83, 84](#)
C:/Users/labuser2/Downloads/System/System/ComponentCreator.h, [84](#)
C:/Users/labuser2/Downloads/System/System/ConcreteRocketBuilder.cpp, [85](#)
C:/Users/labuser2/Downloads/System/System/ConcreteRocketBuilder.h, [85, 86](#)
C:/Users/labuser2/Downloads/System/System/CoreCreator.cpp, [86](#)
C:/Users/labuser2/Downloads/System/System/CoreCreator.h, [87](#)
C:/Users/labuser2/Downloads/System/System/CrewDragon.cpp, [87](#)
C:/Users/labuser2/Downloads/System/System/CrewDragon.h, [87, 88](#)
C:/Users/labuser2/Downloads/System/System/Director.cpp, [88](#)
C:/Users/labuser2/Downloads/System/System/Director.h, [88, 89](#)
C:/Users/labuser2/Downloads/System/System/Facade.cpp, [89](#)
C:/Users/labuser2/Downloads/System/System/Facade.h, [89, 90](#)
C:/Users/labuser2/Downloads/System/System/Fairing.cpp, [91](#)
C:/Users/labuser2/Downloads/System/System/Fairing.h, [91](#)
C:/Users/labuser2/Downloads/System/System/FalconCore.cpp, [91](#)

[92](#)
 C:/Users/labuser2/Downloads/System/System/FalconCore.cpp, [92](#)
 C:/Users/labuser2/Downloads/System/System/main.cpp, [93](#)
 C:/Users/labuser2/Downloads/System/System/mainpage.h, [93](#)
 C:/Users/labuser2/Downloads/System/System/Mediator.h, [93](#)
 C:/Users/labuser2/Downloads/System/System/Memento.cpp, [94](#)
 C:/Users/labuser2/Downloads/System/System/Memento.h, [94](#)
 C:/Users/labuser2/Downloads/System/System/MerlinEngine.cpp, [95](#)
 C:/Users/labuser2/Downloads/System/System/MerlinEngine.h, [95](#)
 C:/Users/labuser2/Downloads/System/System/MerlinEngineCreator.cpp, [95](#)
 C:/Users/labuser2/Downloads/System/System/MerlinEngineCreator.h, [96](#)
 C:/Users/labuser2/Downloads/System/System/Observer.h, [96, 97](#)
 C:/Users/labuser2/Downloads/System/System/Offline.cpp, [97](#)
 C:/Users/labuser2/Downloads/System/System/Offline.h, [97, 98](#)
 C:/Users/labuser2/Downloads/System/System/Online.cpp, [98](#)
 C:/Users/labuser2/Downloads/System/System/Online.h, [98](#)
 C:/Users/labuser2/Downloads/System/System/RocketCapsule.cpp, [99](#)
 C:/Users/labuser2/Downloads/System/System/RocketCapsule.h, [99](#)
 C:/Users/labuser2/Downloads/System/System/Satellite.cpp, [100](#)
 C:/Users/labuser2/Downloads/System/System/Satellite.h, [100, 101](#)
 C:/Users/labuser2/Downloads/System/System/SatelliteCreator.cpp, [101](#)
 C:/Users/labuser2/Downloads/System/System/SatelliteCreator.h, [102](#)
 C:/Users/labuser2/Downloads/System/System/SatelliteState.h, [102, 103](#)
 C:/Users/labuser2/Downloads/System/System/Simulation.cpp, [103](#)
 C:/Users/labuser2/Downloads/System/System/Simulation.h, [104](#)
 C:/Users/labuser2/Downloads/System/System/SimulationState.cpp, [105](#)
 C:/Users/labuser2/Downloads/System/System/SimulationState.h, [106](#)
 C:/Users/labuser2/Downloads/System/System/StarlinkCreator.cpp, [107](#)
 C:/Users/labuser2/Downloads/System/System/StarlinkCreator.h, [107](#)
 C:/Users/labuser2/Downloads/System/System/StarlinkSatellite.cpp, [108](#)
 C:/Users/labuser2/Downloads/System/System/StarlinkSatellite.h, [108](#)
 C:/Users/labuser2/Downloads/System/System/User.cpp, [108](#)
 C:/Users/labuser2/Downloads/System/System/User.h, [109](#)
 C:/Users/labuser2/Downloads/System/System/VacuumMerlinEngine.cpp, [109](#)
 C:/Users/labuser2/Downloads/System/System/VacuumMerlinEngine.h, [110](#)
 C:/Users/labuser2/Downloads/System/System/VacuumMerlinEngineCreator.cpp, [110, 111](#)
 C:/Users/labuser2/Downloads/System/System/VacuumMerlinEngineCreator.h, [110, 111](#)
 CapsuleArriving, [13](#)
 getState, [13](#)
 handleChange, [14](#)
 CapsuleDeparting, [14](#)
 getStop, [14](#)
 handleChange, [14](#)
 CapsuleDocked, [15](#)
 getState, [15](#)
 handleChange, [15](#)
 CapsuleOffline, [16](#)
 getState, [16](#)
 handleChange, [16](#)
 CapsuleState, [16](#)
 getState, [17](#)
 handleChange, [17](#)
 capsuleType
 RocketCapsule, [51](#)
 Caretaker, [17](#)
 getSize, [18](#)
 retrieveMemento, [18](#)
 storeMemento, [18](#)
 CargoDragon, [18](#)
 CargoDragon, [19](#)
 simulate, [19](#)
 test, [19](#)
 changed
 Satellite, [53](#)
 changeSatelliteState
 Simulation, [59](#)
 clone
 Satellite, [53](#)
 StarlinkSatellite, [69](#)
 colleagueList
 CommNetwork, [21](#)
 CommNetwork, [20](#)
 colleagueList, [21](#)
 notify, [20](#)
 sendMessage, [20](#)
 Component, [21](#)
 add, [22](#)
 Component, [22](#)
 cost, [24](#)
 fireMerlin, [22](#)
 fireVacuumMerlin, [22](#)
 Component, [22](#)

- getCost, 23
- getSize, 23
- land, 23
- remove, 23
- separate, 24
- simulate, 24
- test, 24
- ComponentComposite, 25
 - add, 25
 - fireMerlin, 25
 - getComponent, 26
 - getSize, 26
 - land, 26
 - remove, 26
 - separate, 27
 - simulate, 27
 - test, 27
- ComponentCreator, 27
 - factoryMethod, 28
- ConcreteRocketBuilder, 28
 - buildFalcon9, 28
 - buildFalconHeavy, 28
 - constructCapsule, 29
 - createSimulation, 29
 - getCapsule, 29
 - getSpacecraft, 29
- construct
 - Director, 33
- constructCapsule
 - Builder, 12
 - ConcreteRocketBuilder, 29
 - Director, 33
- containsCall
 - Simulation, 60
- CoreCreator, 30
 - factoryMethod, 30
- cost
 - Component, 24
- createMemento
 - Simulation, 60
- createSimulation
 - Builder, 13
 - ConcreteRocketBuilder, 29
 - Director, 33
- CrewDragon, 30
 - CrewDragon, 31
 - getPassengers, 31
 - setPassengers, 31
 - simulate, 31
 - test, 32
- deliverCrew
 - Facade, 34
 - Simulation, 60
- detach
 - Satellite, 53
- Director, 32
 - construct, 33
 - constructCapsule, 33
 - createSimulation, 33
 - Director, 32
 - distributeSatellites
 - Facade, 34
 - Simulation, 60
 - editSimulation
 - Facade, 35
 - Facade, 33
 - build, 34
 - deliverCrew, 34
 - distributeSatellites, 34
 - editSimulation, 35
 - fireMerlin, 35
 - getRocket, 35
 - jettisonFairing, 35
 - launch, 35
 - printSimulation, 35
 - retrieveAll, 36
 - retrieveSimulation, 36
 - runSimulation, 36
 - separateBoosters, 36
 - staticFireTest, 36
 - storeSimulation, 36
 - test, 37
 - useCommNetwork, 37
 - factoryMethod
 - ComponentCreator, 28
 - CoreCreator, 30
 - MerlinEngineCreator, 45
 - SatelliteCreator, 56
 - StarlinkCreator, 68
 - VacuumMerlinEngineCreator, 72
 - Fairing, 37
 - Fairing, 38
 - getSatellite, 38
 - getSatellites, 38
 - setSatellites, 38
 - simulate, 39
 - test, 39
 - FalconCore, 39
 - land, 40
 - separate, 40
 - simulate, 40
 - test, 40
 - fireMerlin
 - Component, 22
 - ComponentComposite, 25
 - Facade, 35
 - MerlinEngine, 44
 - Simulation, 60
 - fireVacuumMerlin
 - Component, 22
 - Simulation, 61
 - VacuumMerlinEngine, 71
 - getCapsule
 - ConcreteRocketBuilder, 29

- getCapsuleType
 - SimulationState, 64
- getComponent
 - Component, 22
 - ComponentComposite, 26
- getCost
 - Component, 23
- getID
 - Satellite, 54
- getMethodCalls
 - SimulationState, 65
- getPassengers
 - CrewDragon, 31
 - SimulationState, 65
- getPayloadWeight
 - RocketCapsule, 49
 - SimulationState, 65
- getRocket
 - Facade, 35
- getRocketType
 - SimulationState, 65
- getSatellite
 - Fairing, 38
 - RocketCapsule, 49
- getSatellites
 - Fairing, 38
 - SimulationState, 65
- getSimulationSize
 - Simulation, 61
- getSize
 - Caretaker, 18
 - Component, 23
 - ComponentComposite, 26
- getSpacecraft
 - Builder, 13
 - ConcreteRocketBuilder, 29
- getState
 - CapsuleArriving, 13
 - CapsuleDeparting, 14
 - CapsuleDocked, 15
 - CapsuleOffline, 16
 - CapsuleState, 17
 - Memento, 43
 - RocketCapsule, 49
 - Satellite, 54
 - Simulation, 61
 - StarlinkSatellite, 69
- getType
 - Broadcasting, 11
 - Offline, 46
 - Online, 47
 - SatelliteState, 57
- handleChange
 - Broadcasting, 11
 - CapsuleArriving, 14
 - CapsuleDeparting, 14
 - CapsuleDocked, 15
 - CapsuleOffline, 16
- CapsuleState, 17
 - Offline, 46
 - Online, 47
 - SatelliteState, 57
- ID
 - Satellite, 55
- jettisonFairing
 - Facade, 35
 - Simulation, 61
- land
 - Component, 23
 - ComponentComposite, 26
 - FalconCore, 40
- landBoosters
 - Simulation, 61
- launch
 - Facade, 35
 - Simulation, 61
- Mediator, 41
 - notify, 41
 - sendMessage, 41
- mediator
 - Satellite, 56
- Memento, 43
 - getState, 43
 - setState, 43
- MerlinEngine, 44
 - fireMerlin, 44
 - simulate, 44
 - test, 44
- MerlinEngineCreator, 45
 - factoryMethod, 45
- notify
 - CommNetwork, 20
 - Mediator, 41
 - Satellite, 54
- Observer, 45
 - update, 46
- observerList
 - Satellite, 56
- Offline, 46
 - getType, 46
 - handleChange, 46
- Online, 47
 - getType, 47
 - handleChange, 47
- printSimulation
 - Facade, 35
 - Simulation, 62
- receiveMessage
 - Satellite, 54
- remove

- Component, 23
- ComponentComposite, 26
- removeStage
 - Simulation, 62
- requestStateChange
 - RocketCapsule, 49
 - Satellite, 54
- restoreMemento
 - Simulation, 62
- retrieveAll
 - Facade, 36
- retrieveMemento
 - Caretaker, 18
- retrieveSimulation
 - Facade, 36
- rocket
 - Simulation.cpp, 104
- RocketCapsule, 48
 - addCapsule, 48
 - capsuleType, 51
 - getPayloadWeight, 49
 - getSatellite, 49
 - getState, 49
 - requestStateChange, 49
 - setPassengers, 49
 - setPayloadWeight, 50
 - setSatellites, 50
 - setState, 50
 - simulate, 51
 - state, 51
 - test, 51
- runSimulation
 - Facade, 36
 - Simulation, 62
- Satellite, 51
 - attach, 53
 - changed, 53
 - clone, 53
 - detach, 53
 - getID, 54
 - getState, 54
 - ID, 55
 - mediator, 56
 - notify, 54
 - observerList, 56
 - receiveMessage, 54
 - requestStateChange, 54
 - Satellite, 52
 - satelliteState, 56
 - sendMessage, 55
 - setMediator, 55
 - setState, 55
- SatelliteCreator, 56
 - factoryMethod, 56
 - setIDCount, 56
- SatelliteState, 57
 - getType, 57
 - handleChange, 57
- satelliteState
 - Satellite, 56
- sendMessage
 - CommNetwork, 20
 - Mediator, 41
 - Satellite, 55
 - Simulation, 63
- separate
 - Component, 24
 - ComponentComposite, 27
 - FalconCore, 40
- separateBoosters
 - Facade, 36
 - Simulation, 63
- setCapsuleType
 - SimulationState, 65
- setIDCount
 - SatelliteCreator, 56
 - StarlinkCreator, 68
- setMediator
 - Satellite, 55
- setMethodCalls
 - SimulationState, 66
- setPassengers
 - CrewDragon, 31
 - RocketCapsule, 49
 - SimulationState, 66
- setPayloadWeight
 - RocketCapsule, 50
 - SimulationState, 66
- setRocketType
 - SimulationState, 67
- setSatellites
 - Fairing, 38
 - RocketCapsule, 50
 - SimulationState, 67
- setState
 - Memento, 43
 - RocketCapsule, 50
 - Satellite, 55
 - StarlinkSatellite, 70
- simulate
 - CargoDragon, 19
 - Component, 24
 - ComponentComposite, 27
 - CrewDragon, 31
 - Fairing, 39
 - FalconCore, 40
 - MerlinEngine, 44
 - RocketCapsule, 51
 - VacuumMerlinEngine, 71
- Simulation, 58
 - addCall, 59
 - changeSatelliteState, 59
 - containsCall, 60
 - createMemento, 60
 - deliverCrew, 60
 - distributeSatellites, 60

- fireMerlin, [60](#)
- fireVacuumMerlin, [61](#)
- getSimulationSize, [61](#)
- getState, [61](#)
- jettisonFairing, [61](#)
- landBoosters, [61](#)
- launch, [61](#)
- printSimulation, [62](#)
- removeStage, [62](#)
- restoreMemento, [62](#)
- runSimulation, [62](#)
- sendMessage, [63](#)
- separateBoosters, [63](#)
- Simulation, [59](#)
- staticFireTest, [63](#)
- swapStage, [63](#)
- updateSimulationState, [64](#)
- Simulation.cpp
 - rocket, [104](#)
- SimulationState, [64](#)
 - getCapsuleType, [64](#)
 - getMethodCalls, [65](#)
 - getPassengers, [65](#)
 - getPayloadWeight, [65](#)
 - getRocketType, [65](#)
 - getSatellites, [65](#)
 - setCapsuleType, [65](#)
 - setMethodCalls, [66](#)
 - setPassengers, [66](#)
 - setPayloadWeight, [66](#)
 - setRocketType, [67](#)
 - setSatellites, [67](#)
- StarlinkCreator, [67](#)
 - factoryMethod, [68](#)
 - setIDCount, [68](#)
- StarlinkSatellite, [68](#)
 - clone, [69](#)
 - getState, [69](#)
 - setState, [70](#)
 - StarlinkSatellite, [69](#)
- state
 - RocketCapsule, [51](#)
- staticFireTest
 - Facade, [36](#)
 - Simulation, [63](#)
- storeMemento
 - Caretaker, [18](#)
- storeSimulation
 - Facade, [36](#)
- swapStage
 - Simulation, [63](#)
- test
 - CargoDragon, [19](#)
 - Component, [24](#)
 - ComponentComposite, [27](#)
 - CrewDragon, [32](#)
 - Facade, [37](#)
 - Fairing, [39](#)
 - FalconCore, [40](#)
 - MerlinEngine, [44](#)
 - RocketCapsule, [51](#)
 - VacuumMerlinEngine, [72](#)
- update
 - Observer, [46](#)
 - User, [71](#)
- updateSimulationState
 - Simulation, [64](#)
- useCommNetwork
 - Facade, [37](#)
- User, [70](#)
 - update, [71](#)
 - User, [70](#)
- VacuumMerlinEngine, [71](#)
 - fireVacuumMerlin, [71](#)
 - simulate, [71](#)
 - test, [72](#)
- VacuumMerlinEngineCreator, [72](#)
 - factoryMethod, [72](#)