

# Programming with Java Controller

# Programming with Java Controller

# Which Controller?

- There are two Controller!
- `org.matsim.run.Controller`
  - public / Part of API
  - used as entry point to run simulations
  - very simple, delegates most of the work to the other Controller
- `org.matsim.core.controller.Controller`
  - responsible for running iterations / simulations
  - reading input files, configuring features / modules, run iterations
  - modules interact with Controller
- We will look at the latter one today

# Creating a Controller

```
Controller c = new Controller("config.xml");
```

---

```
Config config = Gbl.createConfig(new String[] {"config.xml"});  
// modify configuration  
Controller c = new Controller(config);
```

---

```
Scenario scenario = new ScenarioImpl();  
// modify scenario  
Controller c = new Controller(scenario);
```

---

**Not yet available.**

**Available within weeks.**

```
c.run(); // runs the simulation
```

# Configuring Controller

All configuration must be done before `run()` is called!

```
Controller.setOverwriteFiles(true);
```

a very dangerous option, but the most required one

```
Controller.setCreateGraphs(false);
```

many analyses output graphs, but not all people are interested in it

```
Controller.setWriteEventsInterval(5);
```

events will only be written every 5th iteration, mostly for performance reasons

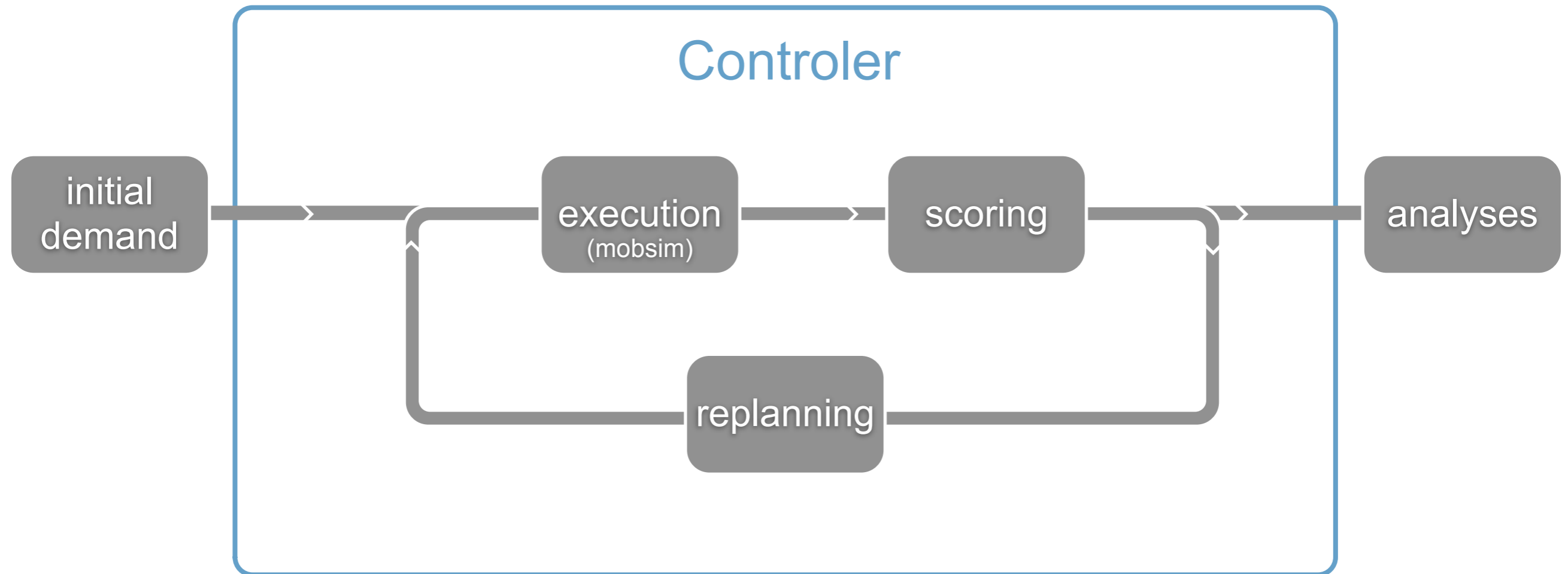
```
Controller.setWriteEventsInterval(0)
```

 disables writing of events completely

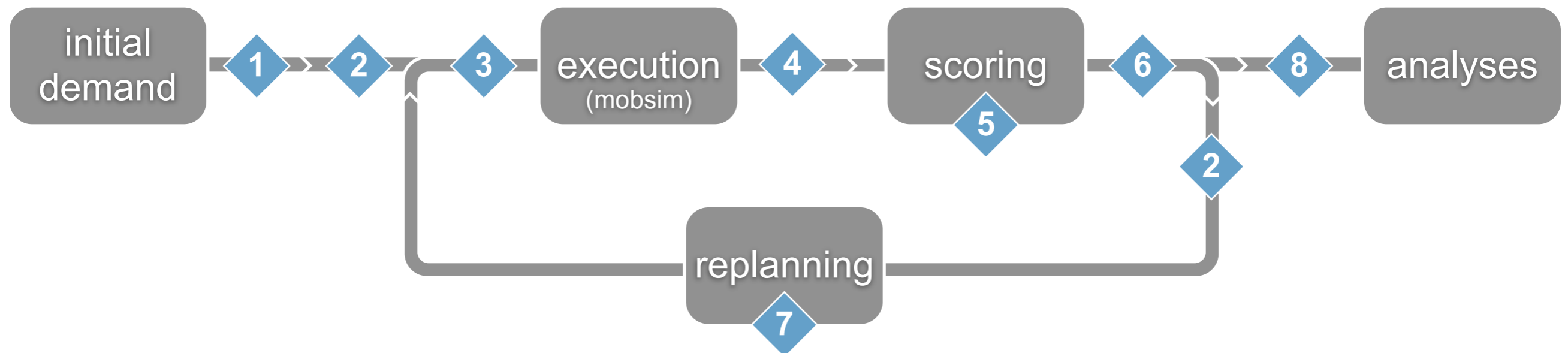
# Real-world example

```
public class MyController {  
    public static void main(String[] args) {  
        Controller c = new Controller(args);  
        c.setOverwriteFiles(false);  
        c.setCreateGraphs(false);  
        c.setWriteEventsInterval(10);  
        c.run();  
    }  
}
```

# MATSim Lifecycle



# Extension Points



## Controler Events:

- |  |                                       |
|--|---------------------------------------|
| <b>1</b> Simulation Starts ("Startup") | <b>5</b> Scoring                      |
| <b>2</b> Iteration Starts              | <b>6</b> Iteration Ends               |
| <b>3</b> Before Mobsim                 | <b>7</b> Replanning                   |
| <b>4</b> After Mobsim                  | <b>8</b> Simulation Ends ("Shutdown") |

Custom code can be executed at each of the extension points.

# ControllerEvents and ControllerListeners

For each extension point, specific ControllerEvents and ControllerListeners are available:

```
org.matsim.core.controller.events.*
```

```
org.matsim.core.controller.listener.*
```

E.g.:

```
IterationStartsEvent, IterationStartsListener
```

```
IterationBeginsEvent, IterationBeginsListener
```

# Example Code

```
import org.matsim.core.controller.events.*;
import org.matsim.core.controller.listener.*;

public class MyIterationEndsListener implements IterationEndsListener {
    public void notifyIterationEnds(IterationEndsEvent event) {
        System.out.println("iteration " + event.getIteration()
            + " / " + event.getController().getLastIteration());
    }
}
```

---

```
public class MyController {
    public static void main(String[] args) {
        Controller c = new Controller(args);
        c.addControllerListener(new MyIterationEndsListener());
        c.run();
    }
}
```

# Combining EventHandler and ControllerListener

```
public class MyEventHandler implements ... {  
    ...  
    public void printStatistics() { ... }  
}
```

---

```
public class MyControllerListener  
    implements StartupListener, IterationEndsListener {  
    private MyEventHandler eh = new MyEventHandler();  
    public void notifyStartup(StartupEvent event) {  
        event.getController().getEvents().addHandler(this.eh);  
    }  
    public void notifyIterationEnds(IterationEndsEvent event) {  
        this.eh.printStatistics();  
    }  
}
```

# Code-Stability of Controller

- Controller, ControllerEvents, ControllerListeners are not yet in public API (org.matsim.api)
- These classes and their methods can change their location, name, arguments; methods could even disappear completely
- If your work heavily depends on the presented features / classes, please speak to us about getting commit-access to the repository

# Experiments / DIY

- How much faster is the simulation if no events are written?
- What graphs are automatically generated?
- Try to print out a message at the end of each iteration
- Count all events and print out the number after the mobsim ends
- Try to automatically load your custom Events-Handler and write out some analysis-information at the end of an iteration, or at the end of the complete simulation
- Before the mobsim starts, calculate the average number of plans an agent has (hints: `BeforeMobsimListener`, `event.getController.getPopulation()` ...)
- Calculate the total distance all agents travel during one iteration (hints: `LinkEnterEventHandler`, `IterationEndsListener`)

- Try the experiments
- Ask us questions, we're here for answering them
- This is the last session, you can ask also about earlier sessions, we will *try* to answer those questions as well
- A few final, closing words will be spoken before lunch