MATSim API

Marcel Rieser

Simunto GmbH

MATSim Training Volkswagen AG
April / May, 2021

MATSim API

```
public static void main(String[] args) {
  Config config = ConfigUtils.loadConfig(configFilename);
  Scenario scenario = ScenarioUtils.loadScenario(config);
  Controler controller = new Controler(scenario);
  controller.run();
}
```

- Config
- Scenario
- Controller

MATSim API: Config

Create or load a config object:

```
Config config1 = ConfigUtils.createConfig();

Config config2 = ConfigUtils.loadConfig("/path/to/config.xml");
```

Create a default config and write it out:

```
Config config = ConfigUtils.createConfig();
new ConfigWriter(config, ConfigWriter.Verbosity.all).write("defaultConfig.xml");
```

MATSim API: Config

`Config` provides access to the different groups and parameters.

Examples:

```
config.controler().setLastIteration(10);
int eventsInterval = config.controler().getWriteEventsInterval();

config.network().setInputFile("/path/to/network.xml");

config.planCalcScore().getScoringParameters(subpopulation).addActivityParams(...);
config.planCalcScore().getScoringParameters(subpopulation).addModeParams(...);
```

MATSim API: Scenario

Create a scenario, with empty data containers:

```
Scenario scenario = ScenarioUtils.createScenario(config);
```

Create and load a scenario:

```
Scenario scenario = ScenarioUtils.loadScenario(config);
```

This loads all the standard files specified in the configuration (network, population, schedule, ...)

MATSim API: Scenario

`Scenario` provides access to data containers:

```
Network network = scenario.getNetwork();

Population population = scenario.getPopulation();

ActivityFacilities facilities = scenario.getActivityFacilities();

TransitSchedule schedule = scenario.getTransitSchedule();
```

MATSim API: Scenario

By default, a Scenario is immutable.
Its data containers cannot be replaced.

```
scenario.setNetwork(myNetwork); // this will not compile
```

But the content of data containers can be modified.

```
scenario.getNetwork().addLink(myLink); // this works
```

MATSim API: Controller

Create a controller:

```
import org.matsim.core.controler.Controler; // there is another Controler class, use this

Controler controller = new Controler(scenario);
```

Run the simulation:

```
controller.run();
```

MATSim API: Controller

`Controler` provides access to important data structures.

```
Config config = controller.getConfig();
Scenario scenario = controller.getScenario();

EventsManager events = controller.getEvents();

Integer iteration = controller.getIterationNumber(); // might be null in startup or shutdown phase
```

MATSim API: Extending Controller with ControlerListeners

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

public class MyControllerListener implements StartupListener, ShutdownListener, IterationStartsListener {

  @Override
  public void notifyIterationStarts(IterationStartsEvent iterationStartsEvent) {
    ...
  }

  @Override
  public void notifyStartup(StartupEvent startupEvent) {
    ...
  }

  @Override
  public void notifyShutdown(ShutdownEvent shutdownEvent) {
    ...
  }

}
```

```
Controler controller = new Controler(scenario);
controller.addControlerListener(new MyControllerListener());
controller.run();
```

MATSim API: Extending Controller by Dependency Injection

```
import org.matsim.core.controler.AbstractModule;

Controler controller = new Controler(scenario);

controller.addOverridingModule(new AbstractModule() {
  @Override
  public void install() {
    this.addEventHandlerBinding().to(MyEventHandler.class);
    this.addPlanStrategyBinding("MyStrategy").to(MyStrategy.class);
    this.addRoutingModuleBinding("car").to(MyRoutingModule.class);
    this.addMobsimListenerBinding().to(MyMobsimListener.class);

    this.… // there are more methods, use code completion for the full list

    // generic alternative
    this.bind(SomeInterface.class).to(SomeImplementation.class);
  }
});

controller.run();
```

MATSim API: ControlerIO

`ControlerIO` provides standardized access to (output) file paths.
It is available from Controler as well as from ControlerEvents.

```
String filename = controller.getControlerIO().getIterationFilename(iteration, "myOutput.txt");
// write data to 'filename'
```

```
public class MyControllerListener implements IterationEndsListener {

  @Override
  public void notifyIterationEnds(IterationEndsEvent event) {
    String filename = event.getServices()
                           .getControlerIO().getIterationFilename(event.getIteration(), "myOutput.txt");
    // e.g. write collected data to 'filename'
  }
  ...
}
```

MATSim API: IOUtils

MATSim provides a helper class `IOUtils` to work with (gzipped) files.
It automatically (de)compresses gzipped files if the filename ends on `.gz`:

```
String filename = "path/to/data.txt";
String filename = "path/to/data.txt.gz";

try (BufferedWriter writer = IOUtils.getBufferedWriter(filename)) {
  writer.write("Hello MATSim!");
} catch (IOException e) {
  log.error("oops!", e);
}
```

Read data:

```
String filename = "path/to/data.txt";
String filename = "path/to/data.txt.gz";

try (BufferedReader reader = IOUtils.getBufferedReader(filename)) {
  String line = reader.readLine();
  log.info(line);
} catch (IOException e) {
  log.error("oops!", e);
}
```

MATSim API: Data Containers

Use Data Containers (network, population, ...) from a scenario,
or use `*Utils`-classes to create them:

```
Scenario scenario = ScenarioUtils.createScenario(ConfigUtils.createConfig());
Network network = scenario.getNetwork();
Population population = scenario.getPopulation();
```

```
Network network = NetworkUtils.createNetwork();
Population population = PopulationUtils.createPopulation(config);
ActivityFacilities facilities = FacilitiesUtils.createActivityFacilities();

// exception from the *Utils-pattern:
TransitSchedule schedule = new TransitScheduleFactoryImpl().createTransitSchedule();
```

MATSim API: Data Containers

Reading and Writing data containers:

```
Scenario scenario = ScenarioUtils.createScenario(ConfigUtils.createConfig());

Network network = scenario.getNetwork();
Population population = scenario.getPopulation();
ActivityFacilities facilities = scenario.getActivityFacilities();
TransitSchedule schedule = scenario.getTransitSchedule();

new MatsimNetworkReader(network).readFile("/path/to/network.xml.gz");
new NetworkWriter(network).write("/path/to/network.xml.gz");

new PopulationReader(scenario).readFile("/path/to/population.xml.gz");
new PopulationWriter(population).write("/path/to/population.xml.gz");

new MatsimFacilitiesReader(scenario).readFile("/path/to/population.xml.gz");
new FacilitiesWriter(facilities).write("/path/to/population.xml.gz");

new TransitScheduleReader(scenario).readFile("/path/to/transitSchedule.xml.gz");
new TransitScheduleWriter(schedule).writeFile("/path/to/transitSchedule.xml.gz");
```

Sadly, the API is not consistent.

MATSim API: Data Containers, Ids

Elements in data containers have `Id`s for identification.

Ids have a type.

```
Id<Link>    linkId;
Id<Node>    nodeId;
Id<Person>  personId;
```

The type must be specified when creating an Id:

```
Id<Person> personId = Id.create("123", Person.class);

Id<Node> nodeId = Id.create("456", Node.class);

Id<MyClass> myId = Id.create("789", MyClass.class);
```

MATSim API: Data Containers, Factories

Use Factories to modify create new elements for data containers. Each data container provides its own factory.

Don't forget to add the created elements to the data container.

```
NetworkFactory netF = network.getFactory();

Node node1 = netF.createNode(Id.create("1", Node.class), new Coord(100, 300));
Node node2 = netF.createNode(Id.create("2", Node.class), new Coord(200, 400));

network.addNode(node1);
network.addNode(node2);

Link link = netF.createLink(Id.create("1", Link.class), node1, node2);
link.setFreespeed(50.0 / 3.6); // 50 km/h in m/s
link.setLength(300); // meter
link.setCapacity(800); // veh / hour
link.setNumberOfLanes(1);
link.setAllowedModes(CollectionUtils.stringToSet("car,bus"));

network.addLink(link);
```

MATSim API: Data Containers, Factories

```
PopulationFactory popF = population.getFactory();

Person person = popF.createPerson(Id.create("12", Person.class));
Plan plan = popF.createPlan();

Activity act1 = popF.createActivityFromCoord("home", new Coord(150, 250));
act1.setEndTime(8 * 3600);
Leg leg = popF.createLeg("car");
Activity act2 = popF.createActivityFromCoord("work", new Coord(750, 650));

plan.addActivity(act1);
plan.addLeg(leg);
plan.addActivity(act2);

person.addPlan(plan);

population.addPerson(person);
```

MATSim API: Data Containers, Attributes

Attributes can often be stored along objects, or externally using the Id for lookup.

```
// "internal"
person.getAttributes().putAttribute("age", 39);
int age = (Integer) person.getAttributes().getAttribute("age");
```

```
// "external", "ObjectAttributes"
String personId = person.getId().toString();
ObjectAttributes attributes = population.getPersonAttributes();
attributes.putAttribute(personId, "age", 39);
int age = (Integer) attributes.getAttribute(personId, "age");
```

MATSim switches more and more to “internal” attributes.
Know where you store your attributes when accessing them.

MATSim API: Utils classes

```
ConfigUtils
.createConfig(...);
.loadConfig(...);
.applyCommandLine(...);
```

```
ScenarioUtils
.createScenario(...);
.loadScenario(...);
```

```
EventsUtils
.createEventsManager();
```

```
NetworkUtils
.createNetwork();
.createNode(...);
.createLink(...);
.getBoundingBox(...);
.getLinks(... linkIds);
.isMultimodal(...);
.getConnectingLink(node1, node2);
.getCloserNodeOnLink(coord, link);
.getNearestLink(network, coord);
....
```

```
PopulationUtils
.createPopulation(...);
.createPlan(...);
.createActivity(...);
.createLeg(...);
.resetRoutes(...);
.getActivityEndTime(...);
.removeActivity(...);
.removeLeg(...);
.decideOnLinkIdForActivity(...);
.putSubpopulation(...);
....
```

```
PersonUtils
.setCarAvail(...);
.setLicense(...);
.setEmployed(...);
....
```

```
TransitScheduleUtils
.putStopFacilityAttribute(...);
.createQuadTreeOfTransitStopFacilit
....
```

```
FacilitiesUtils
.createActivityFacilities(...);
.setLinkID(...);
.decideOnLinkId(...);
....
```

```
CoordUtils
.plus(...);
.minus(...);
.length(...);
.calcEuclideanDistance(...);
.distancePointLinesegment(...);
....
```

Many contribs also provide their own Utils classes, e.g.:

```
AccessibilityUtils.…
BicycleUtils.…
CarrierUtils.…
EmissionUtils.…
....
```

MATSim API: Time

MATSim internally uses “seconds after midnight”.

To convert to and from human-readable time formats, use the `Time` class:

```
double seconds = Time.parseTime("07:12:35");

String time = Time.writeTime(43200);
```

MATSim does not know days, it just keeps counting the hours (seconds, respectively).

After “23:59:59” comes “24:00:00”, “24:00:01” … “25:00:00” …