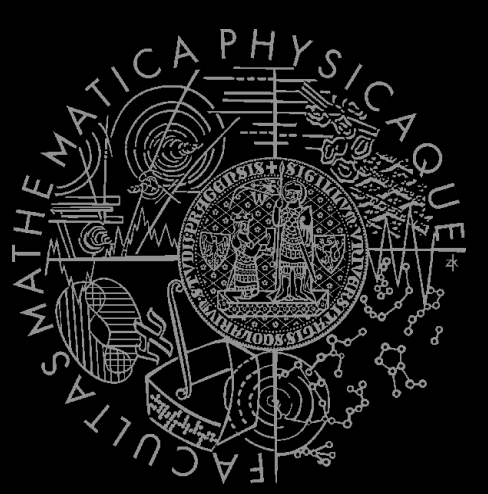
Faculty of mathematics and physics
Charles University at Prague
11th March 2011



UT2004 bots made easy!

Pogamut 3

Lecture 2 – Exploring the map

Warm up!

Fill the test for this lecture!

Home work: ShootBot

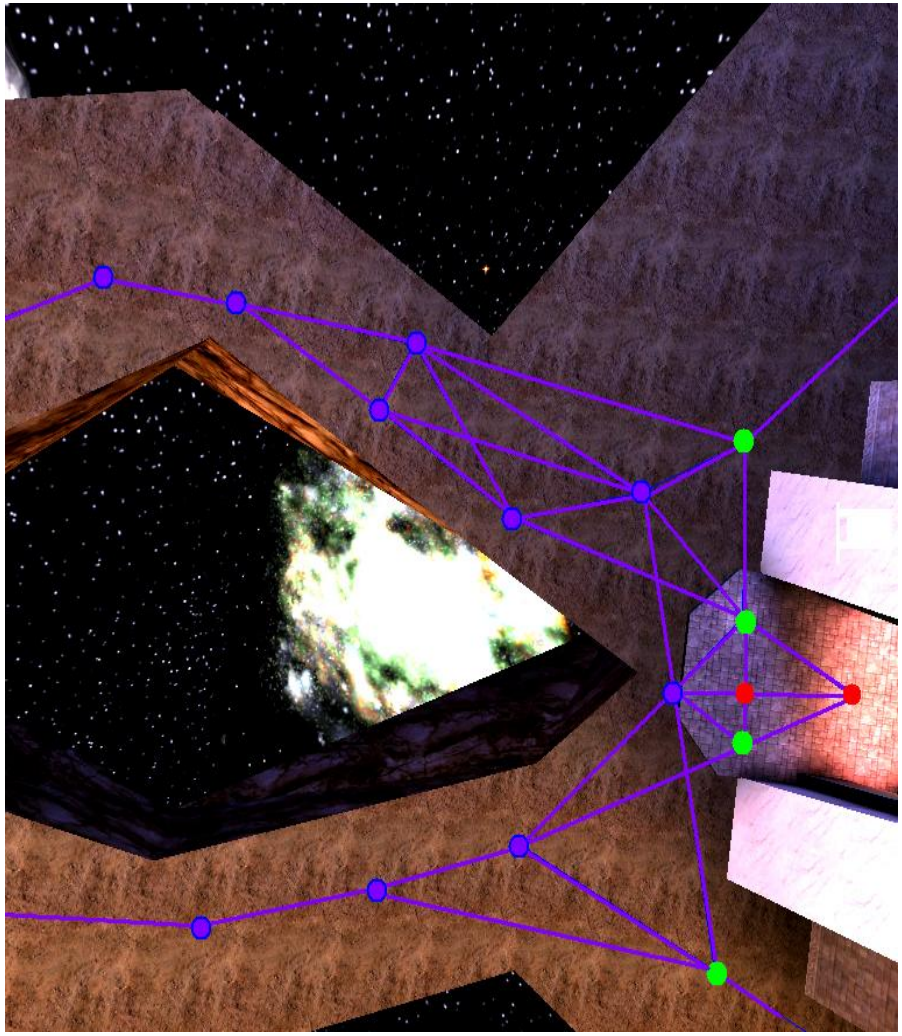
Let's review home works from previous lectures!

Today's menu

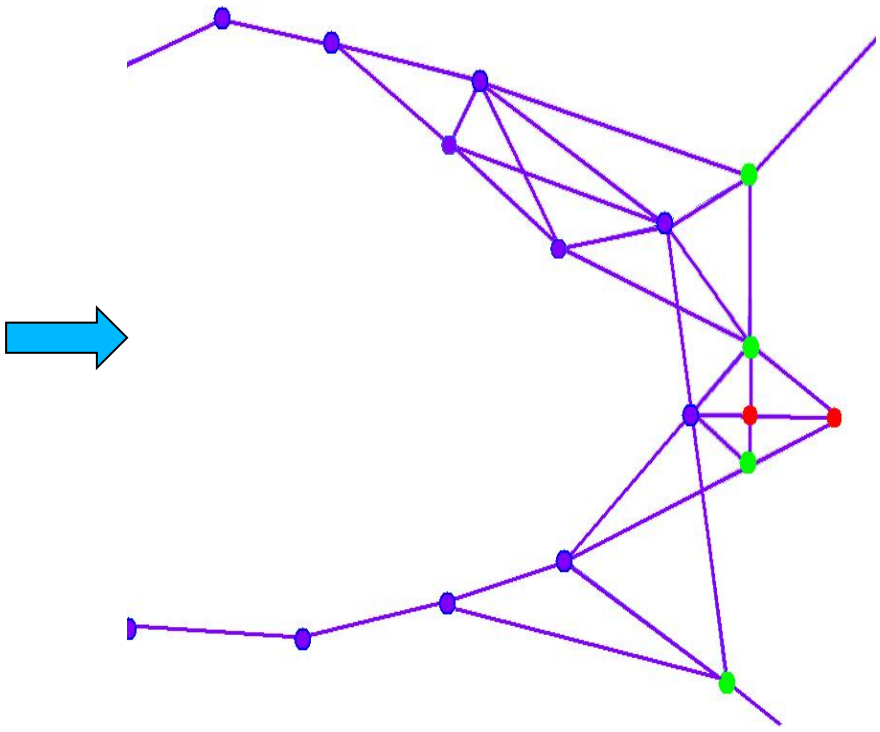
1. **World and its representation**
   - **NavPoints, NavPointNeighbourLink, Items**
2. Event Listeners / Annotations
   - Bot startup sequence
3. World navigation
   - Path executor, Path planner
4. Floyd-Warshal Map
   - NavigationGraphBuilder

UT2004 World Abstraction

Navigation graph



#Navpoints in the map = 100 – cca 5000



UT2004 World Abstraction

Underlaying classes

Classes of interest:

NavPoint, NavPointNeighbourLink, Item

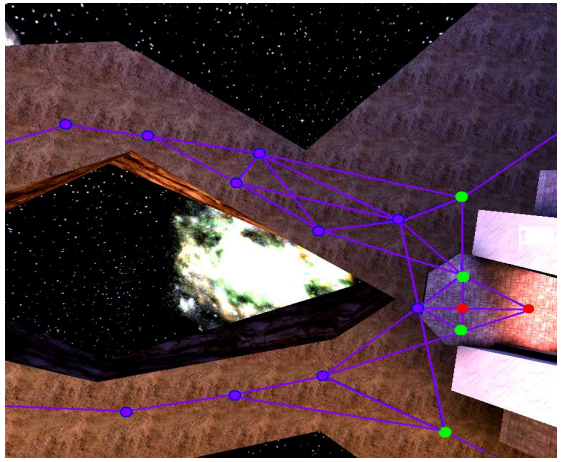
ILocated, Location, DistanceUtils

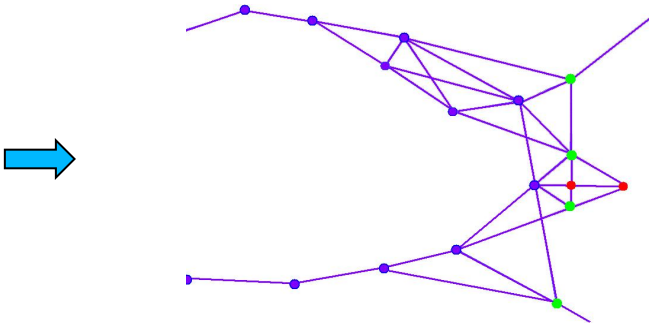
ItemType, ItemType.Category, ItemDescriptor

Methods of interest:

```
this.items.getAllItems(ItemType)
this.descriptors.getDescriptor(ItemType)
this.world.getAll(NavPoint.class)
this.world.getAll(Item.class)
NavPoint.getOutgoingEdges()
NavPoint.getIncomingEdges()
```

!!!





UT2004 World Abstraction

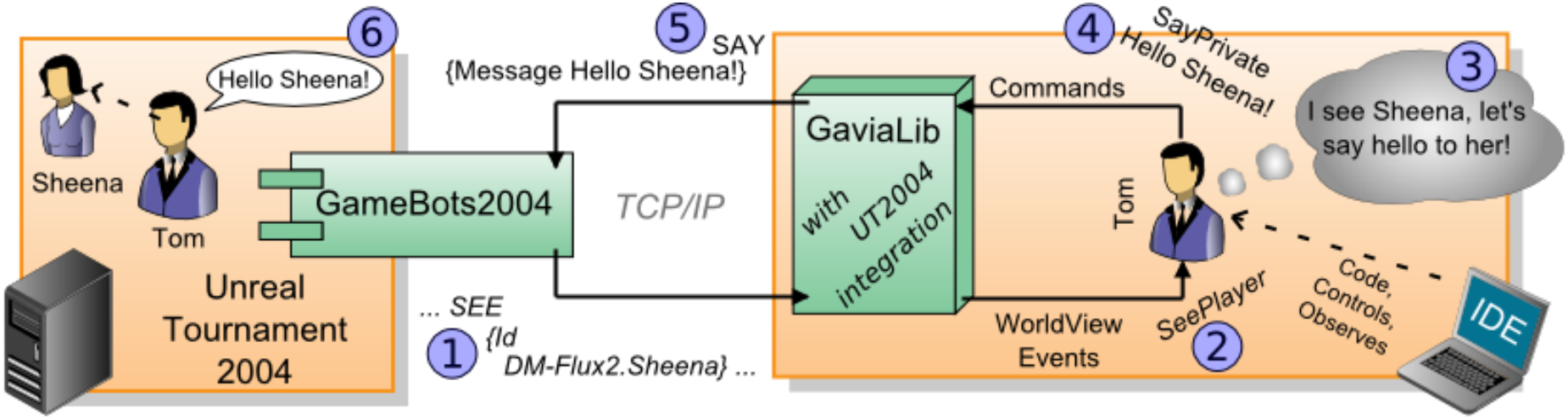
Events, Update Events, Objects, Listeners

- `IWorldEvent` <- `IWorldObjectUpdatedEvent`
- `IWorldObject` <- `NavPoint, Player, Item, …`
  - Every object has `getId()`
- You may hook up listeners to be informed about the events as they happen
- Events (not update events) are not otherwise accessible! Once you miss them, they are gone…

Today's menu

1. World and its representation
   - NavPoints, NavPointNeighbourLink, Items
2. **Event Listeners / Annotations**
   - **Bot startup sequence**
3. Path executor, Path planner
   - Path executor, Path planner
4. Floyd-Warshal Map
   - NavigationGraphBuilder

Events & Listeners



Classes of interest:

`IWorldView <- IVisionWorldView`

`IWorldEventListener`

`IWorldObjectListener`

`IWorldObjectEventListener`

`AnnotationListenerRegistrator`

Methods of interest:

`this.world.addEventListener(…)`

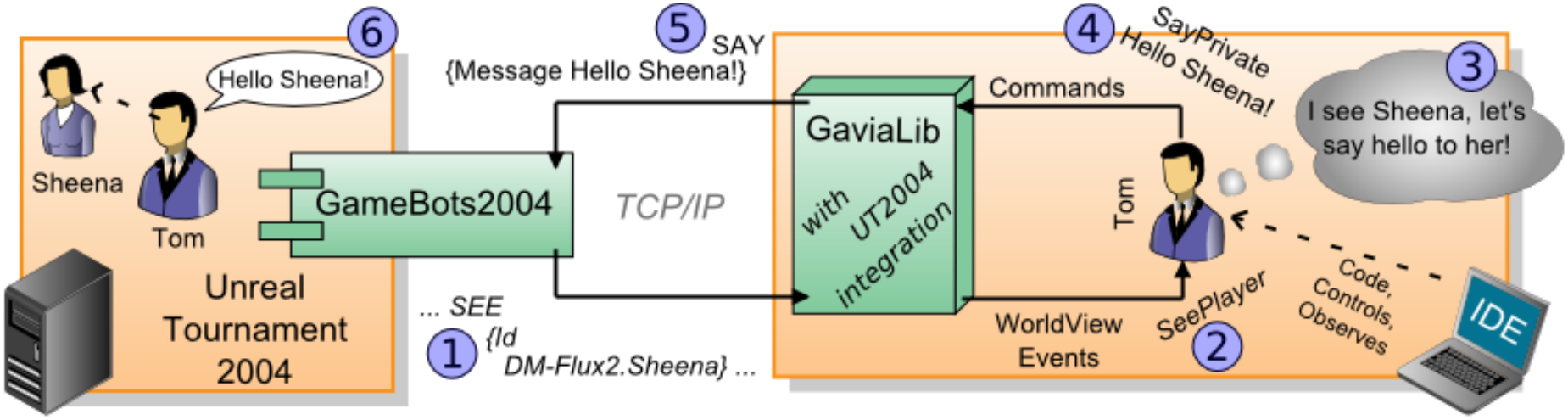
`this.world.addObjectListener(…)`

Method annotations:

`@EventListener`

`@ObjectClassEventListener`

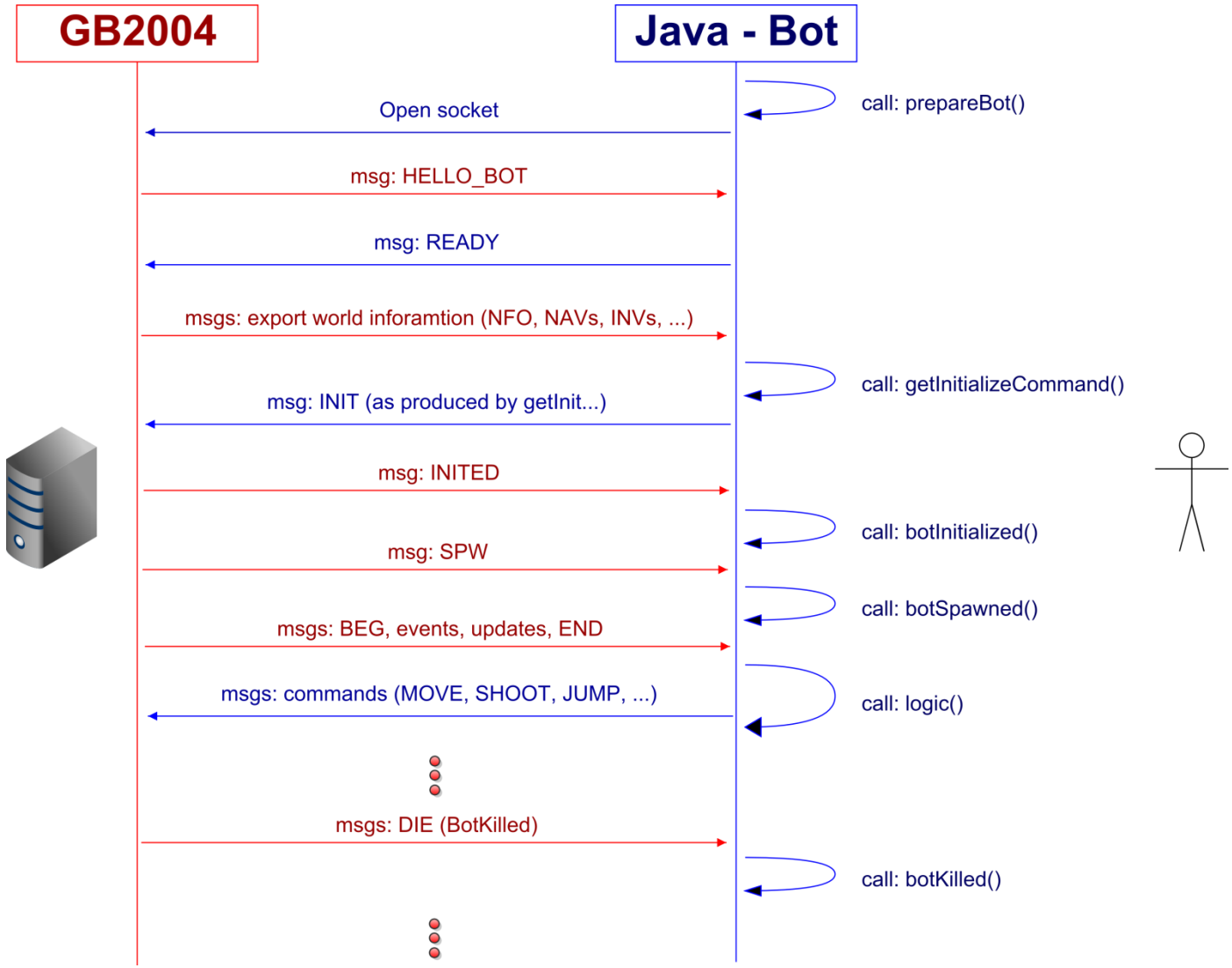
Events & Listeners



- http://diana.ms.mff.cuni.cz/pogamut_files/latest/doc/gamebots/ch06.html
- http://diana.ms.mff.cuni.cz/pogamut_files/latest/doc/javadoc/
  - Search for IWorldEvent and its descendants

Events & Listeners

Bot startup sequence



Events & Listeners

Let's check it out inside ResponsiveBot now!

Assignment 1 (or HomeWork)

- Alter **ResponsiveBot** to create **RetaliatorBot**
  1. If player hits you, remember his ID and try to bite back.
  2. Whenever you deliver at least the same amount of damage to the offender, stop shooting.
  - You must use listeners!
  - What if multiple players hit you at once?

Today's menu

1. World and its representation
   - NavPoints, NavPointNeighbourLink, Items
2. Event Listeners / Annotations
   - Bot startup sequence
3. **World navigation**
   - **Path executor, Path planner**
4. Floyd-Warshal Map
   - NavigationGraphBuilder

World navigation

Step by step

1. Decide where to go
2. Plan the path (list of navpoints)
3. Follow the path
   - Handle jumps&lifts along the way!
     - Do you know right constants?
   - World is non-deterministic, be sure to check how the action is executing!

Don’t worry it’s already wrapped up ☺

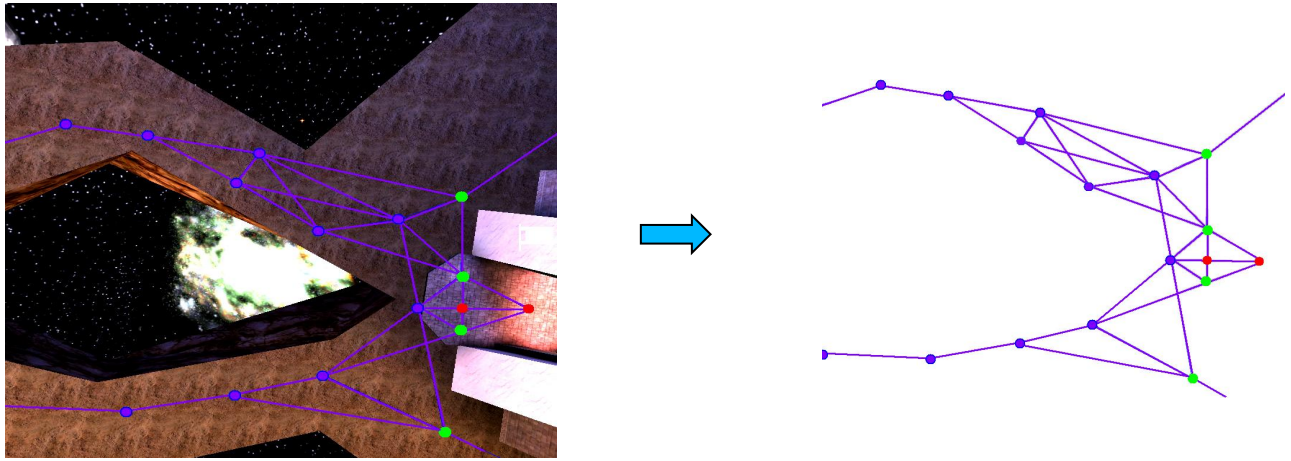
World navigation

Path planner & Path executor

1. Decide where to go (Deliberation!)
   - `items.getSpawnedItems(ItemType)`
   - `DistanceUtils.getNearest(…)`
2. Plan the path (list of navpoints)
   - `pathPlanner.computePath(ILocated from, to)`
3. Follow the path
   - `pathExecutor.followPath(path)`
   - `pathExecutor.isExecuting()`
   - `pathExecutor.addStuckExecutor(…)`

World navigation

NavigationBot



Let's check it out inside NavigationBot now!

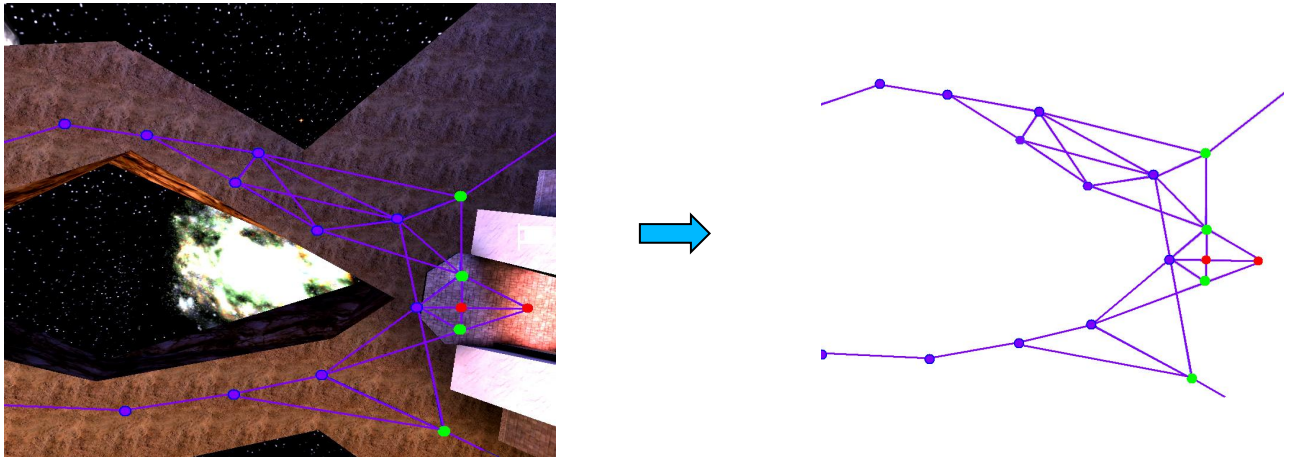
Assignment 2 (or HomeWork)

- Alter **NavigationBot** to create **CollectorBot**
  1. NavigationBot is working via events, reformulate it so the functionality will be implemented in `logic()`
  2. Implement collector behavior – always try to get to the nearest spawned item
  - How to be sure that the bot has arrived to the item's location?
  - What if the location is currently unreachable?
    - See `TabooSet` class

Today's menu

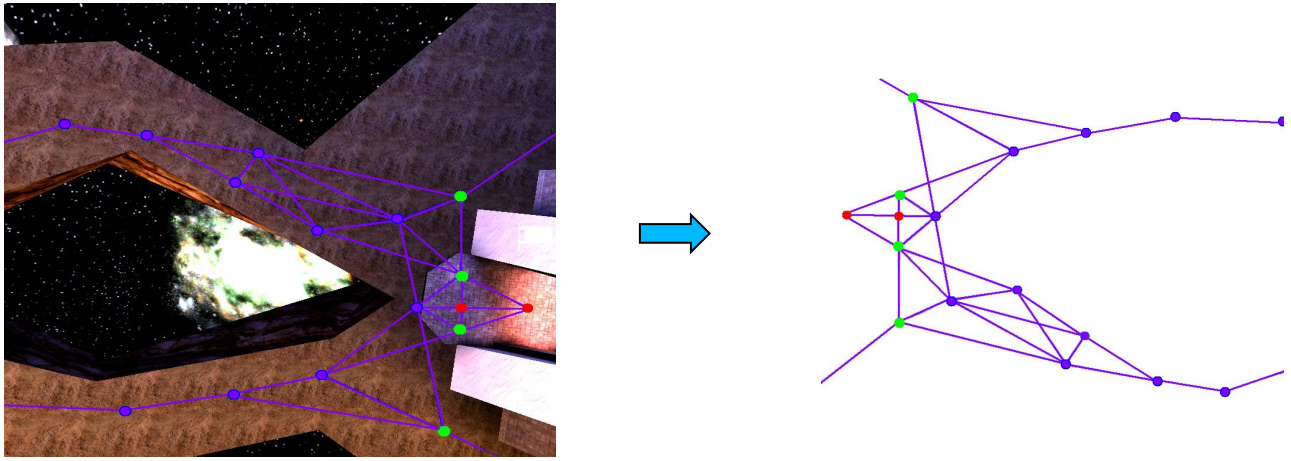
1. World and its representation
   - NavPoints, NavPointNeighbourLink, Items
2. Event Listeners / Annotations
   - Bot startup sequence
3. World navigation
   - Path executor, Path planner
4. **Floyd-Warshal Map**
   - **NavigationGraphBuilder**

Navigation graph



- There are up to 5000 nav-points per map.
- Floyd-Warshall algorithm
  - O(n^3) ~ 125.000.000.000 "operations" ~ 2 mins
- `fwMap.computePath(NavPoint from, to)`
- `info.getNearestNavPoint()`
- `Item.getNavPoint()`

Navigation graph
Navigation graph builder



- We may even alter the navigation graph by hand!
- `navBuilder.removeEdge(…)`
- `navBuilder.removeEdgesBetween(…)`
- `navBuilder.newNavpoint(…)`
- Where (in which method) would you configure it?

fwMap vs. pathPlanner

`fwMap`

- Path precomputed
- Graph may be altered
- Can't plan to all locations

`pathPlanner`

- Path is planned at UT2004
- Graph is fixed
- May plan everywhere

`pathExecutor` works with both!

Assignment 3 (or HomeWork)

- Combine **RetaliatorBot** and **CollectorBot**
  1. Run around and collect items.
  2. If you're hit by a player, bite back (switch to follow-shoot bot behavior until required amount of damage is dealt).
  3. If bot has < 100 healths, prioritize `ItemType.Category.HEALTH` items.
  4. Solve problem with temporary unreachable items (you bot must not get stuck).

Send your assignments to

- Completely zip-up your project(s) folder
- Send it to:
  - Jakub Gemrot (Friday practice lessons)
    - jakub.gemrot@gmail.com
  - Michal Bída (Wednesday practice lessons)
    - michal.bida@gmail.com