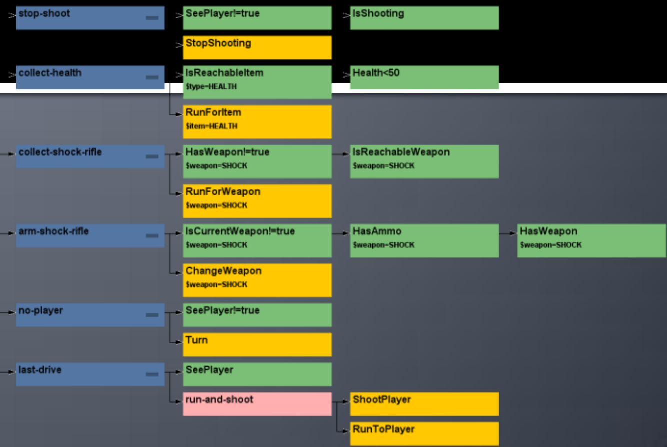
Faculty of Mathematics and Physics
Charles University in Prague
28th April 2014

UT2004 bots made easy!

Pogamut 3

Lecture 8 – BOD, yaPOSH & DeathMatch



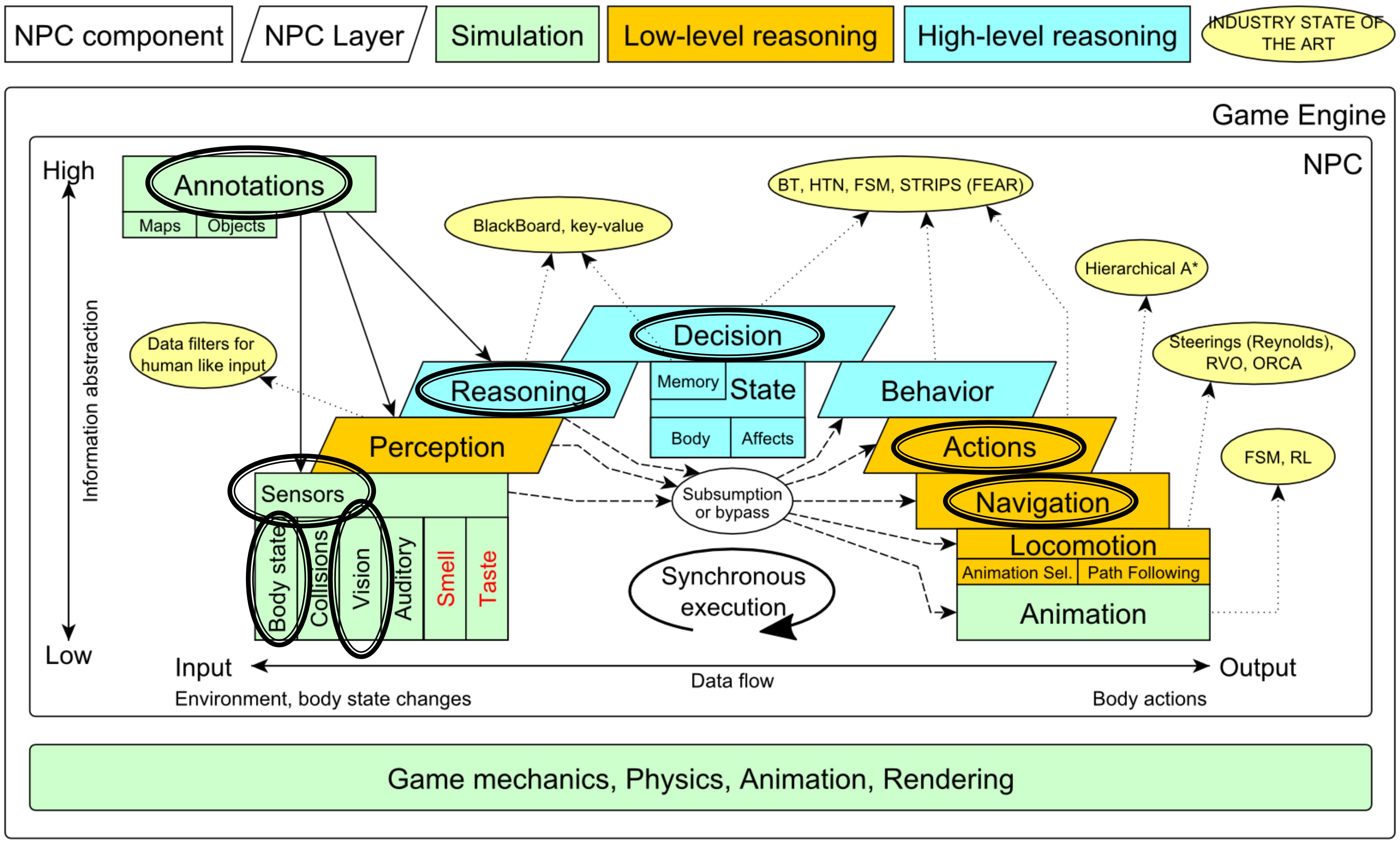
Warm Up!

- Fill the short test for this lessons
  - 6 minutes limit
  - http://alturl.com/3dbrg
  - https://docs.google.com/forms/d/1lzS1RxHrMcRE-Ni8CeaxCTcdtgCqRfSZDikBz92fdbI/viewform

Today's menu

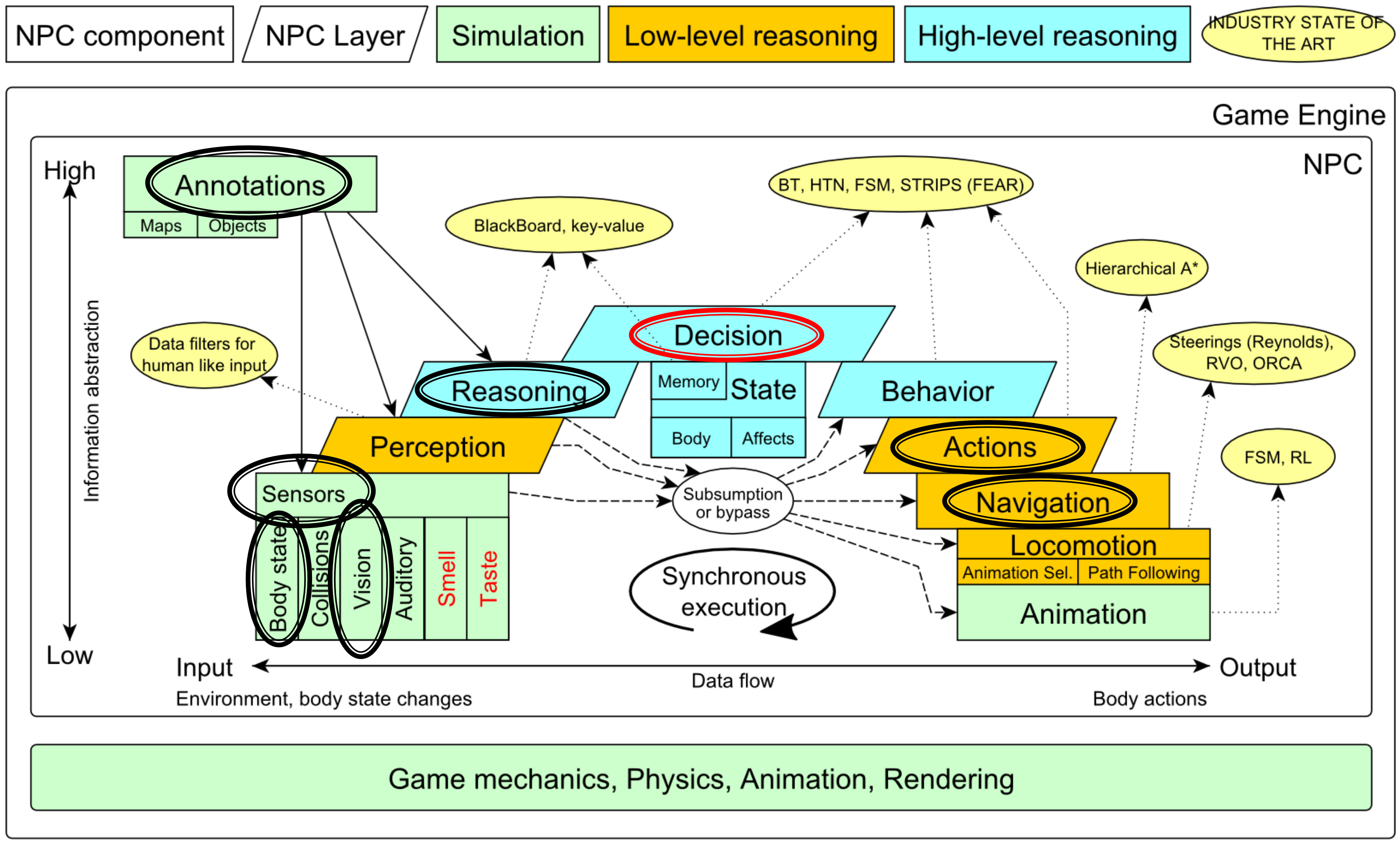
1. **Big Picture**
2. BOD (Behavior Oriented Design)
3. Gentle POSH introduction
4. DeathMatch Bot

Big Picture
Already covered



Big Picture

Today



Today's menu

1. Big Picture
2. **BOD (Behavior Oriented Design)**
3. Gentle POSH introduction
4. DeathMatch Bot

Behavior Oriented Design
Methodology

- **BOD** (Behavior Oriented Design)
  - A methodology for developing control of complex intelligent agents
    - virtual reality characters, humanoid robots or intelligent environments…
- Combines the advantages of Behavior-Based AI and Object Oriented Design.
- Authored by Joanna J. Bryson
  - http://www.cs.bath.ac.uk/~jjb/web/bod.html

How to think?
Intelligence by design

Behavior Oriented Design

by Joanna J. Bryson (UK)

http://www.cs.bath.ac.uk/~jjb/web/bod.html

1. Specify top-level decision
   a) Name the behaviors that the bot should do
   b) Identify the list of sensors that is required to perform the behavior
   c) Identify the priorities of behaviors
   d) Identify behavior switching conditions
2. Recursion on respective behaviors until primitive actions reached

Behavior Oriented Design

BOD in human language

1. State the goal of you agent behavior
   - E.g. It will be a Deathmatch bot
2. Brainstorm what it will mean to fulfill the behavior goal
   - E.g. fight players, gather items
3. Think about conditions that should be fulfilled for the respective behaviors
   - E.g. I'll fight only when I see enemy and have proper weapon
4. Revise, revise, revise
   - Oh wait, what if I don't have the proper weapon, I should add a behavior to flee from fight and gather some weapon.
5. Pick one of the specified top level behaviors and apply recursion from point 1!
6. When you end up with sufficiently simple and clear defined sense/action – **NAME IT WELL**, implement it and test it!

Behavior Oriented Design
Iterative Development

Recursion == Iterative development

1. Select a part of the plan to extend next.
2. Extend the agent with that implementation
   - Extend the plan, code actions and senses
   - Test and debug that code (!!!)
3. Revise the current specification.

Behavior Oriented Design

Revising BOD Specifications

- Name the behaviors (functions) logically!
  - Good method name is better than documentation!
- Reduce code redundancy
  - Use copy paste with caution or not at all!
- *Avoid Complex Conditions*
  - The shorter condition, the better the understanding
- *Avoid Too Many If-then rules at one level*
  - One level of decision making usually needs no more than 5 to 7 if-then rules, they may contain fewer..
- *When in doubt, favor simplicity.*

Practice Lesson

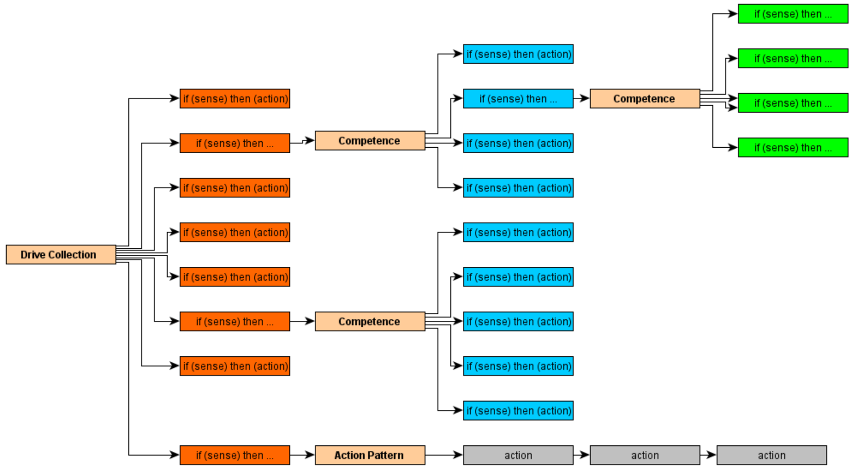
Outline

1. Big Picture
2. BOD (Behavior Oriented Design)
3. **Gentle POSH introduction**
4. DeathMatch Bot

yaPOSH
Introduction

- **yaPOSH**
  - **y**et-**a**nother **P**arallel-rooted, **O**rdered **S**lip-stack **H**ierarchical planner
- To put it simply:
  - a reactive planner working with **FIXED**, **PRE-SET** plans
- To put it even simpler:
  - a tool enabling to specify **if – then** rules with **priority** in a **tree like structure**
- Advantage:
  - Makes you think about the behavior in human terms more than the code

yaPOSH
Primitives



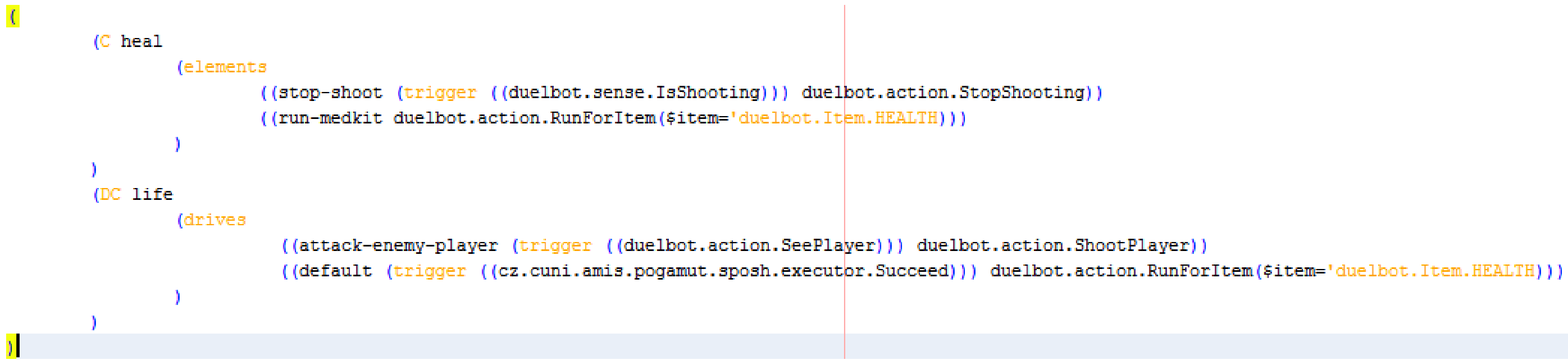
- **Actions and Senses**
  - if (sense) then (action)
- **Drive Collection (DC)**
  - First level of if-then rules
- **Competence (C)**
  - Second – Nth level of if-then rules
- **Action Patterns (AP)**
  - Specifies N actions that will be performed in a sequence

yaPOSH
Plan structure (Java glasses)

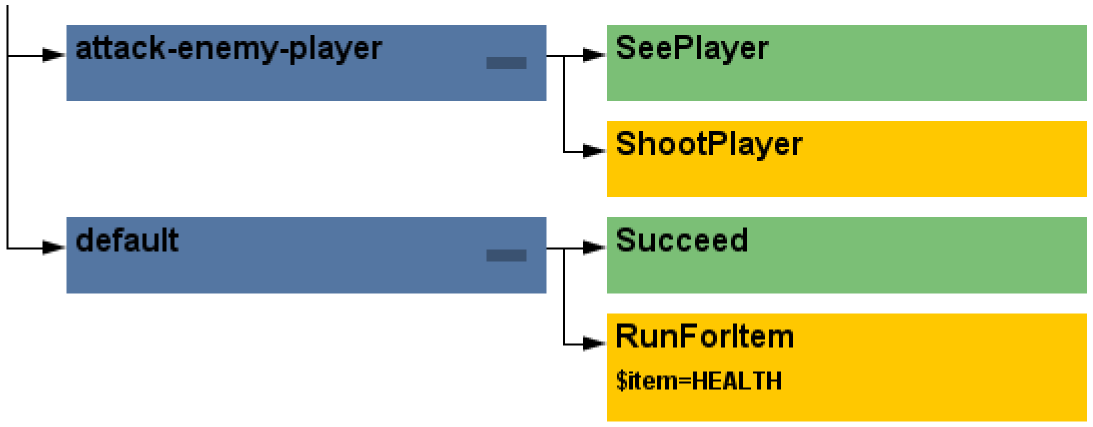
```
DriveCollection(
    1. if (sense1()) then competence1(); return;
    2. if (sense2()) then competence2(); return;
    3. if (sense3()) then action-pattern1(); return;
    4. if (sense4()) then competence3(); (
           1. if (sense5()) then action1(); return;
           2. if (sense6()) then competence4(); return;
           3. if (sense7()) then action2(); return;
           4. if (sense8()) then action-pattern(); return;
           5. return;
    )
)

ActionPattern(
    while (!action1-finished()) {action1();};
    while (!action2-finished()) {action2();};
    while (!action3-finished()) {action3();};
)
```

yaPOSH

Plan structure (the real)

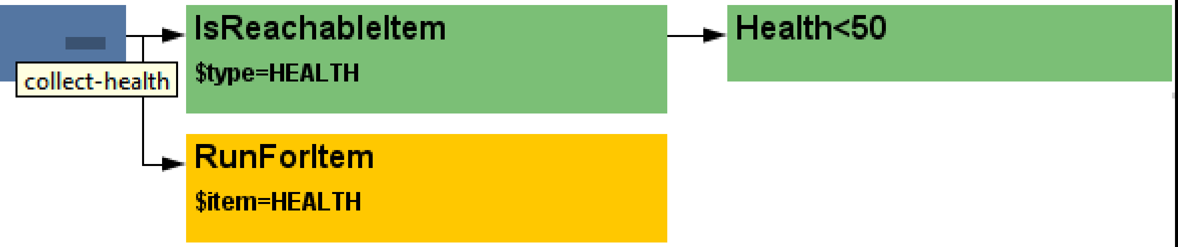


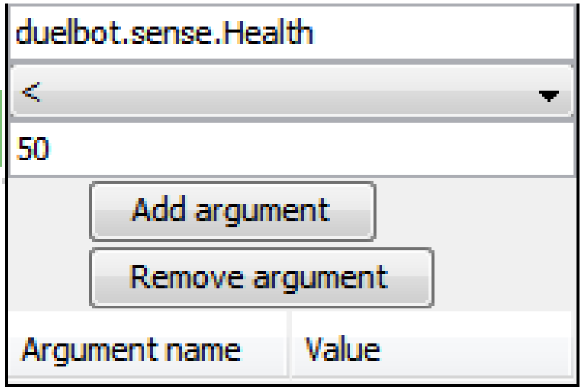
```
(
        (C heal
                (elements
                        ((stop-shoot (trigger ((duelbot.sense.IsShooting))) duelbot.action.StopShooting))
                        ((run-medkit duelbot.action.RunForItem($item='duelbot.Item.HEALTH')))
                )
        )
        (DC life
                (drives
                          ((attack-enemy-player (trigger ((duelbot.action.SeePlayer))) duelbot.action.ShootPlayer))
                          ((default (trigger ((cz.cuni.amis.pogamut.sposh.executor.Succeed))) duelbot.action.RunForItem($item='duelbot.Item.HEALTH')))
                )
        )
)
```



yaPOSH
Senses

- Senses
  - Represent condition (Do I see a player?)
  - Return basic types
    - Boolean, Integer, Double, String, …
  - Can be queried either as **==**, **!=**, **>**, **<**, **<=** or **>=**
  - E.g.:




duelbot.sense.Health

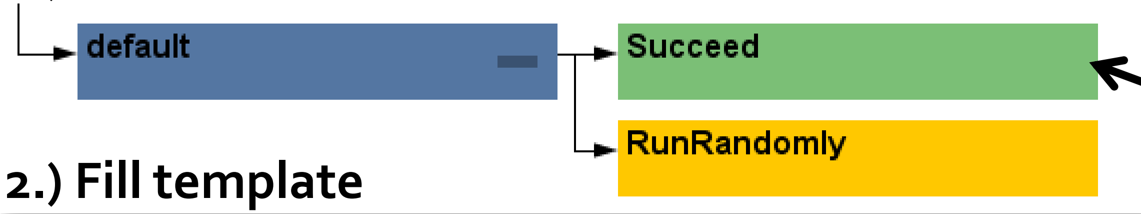
<

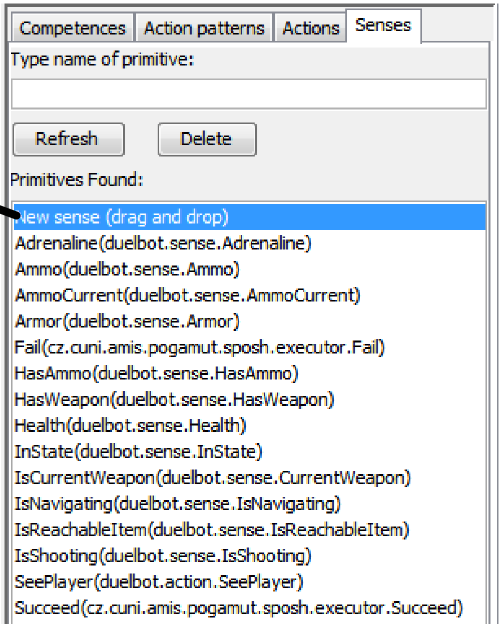
50

yaPOSH

How to make new Sense?

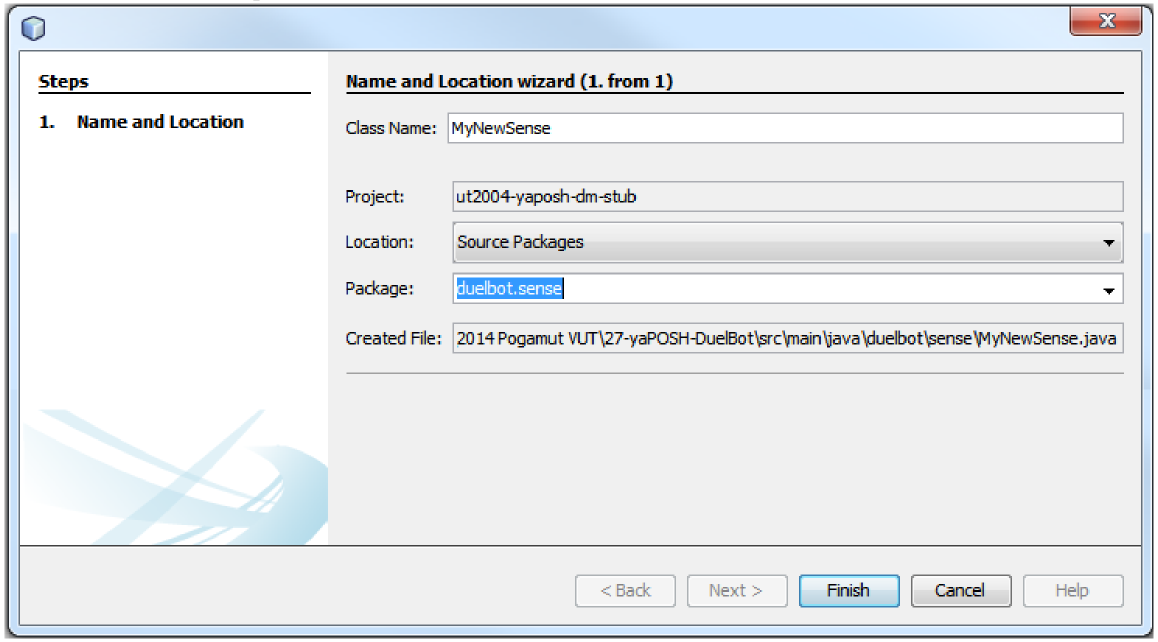
1.) DRAG & DROP!





New sense (drag and drop)
Adrenaline(duelbot.sense.Adrenaline)
Ammo(duelbot.sense.Ammo)
AmmoCurrent(duelbot.sense.AmmoCurrent)
Armor(duelbot.sense.Armor)
Fail(cz.cuni.amis.pogamut.sposh.executor.Fail)
HasAmmo(duelbot.sense.HasAmmo)
HasWeapon(duelbot.sense.HasWeapon)
Health(duelbot.sense.Health)
InState(duelbot.sense.InState)
IsCurrentWeapon(duelbot.sense.CurrentWeapon)
IsNavigating(duelbot.sense.IsNavigating)
IsReachableItem(duelbot.sense.IsReachableItem)
IsShooting(duelbot.sense.IsShooting)
SeePlayer(duelbot.action.SeePlayer)
Succeed(cz.cuni.amis.pogamut.sposh.executor.Succeed)

2.) Fill template



Name and Location wizard (1. from 1)

Class Name: MyNewSense

Project: ut2004-yaposh-dm-stub

Location: Source Packages

Package: duelbot.sense

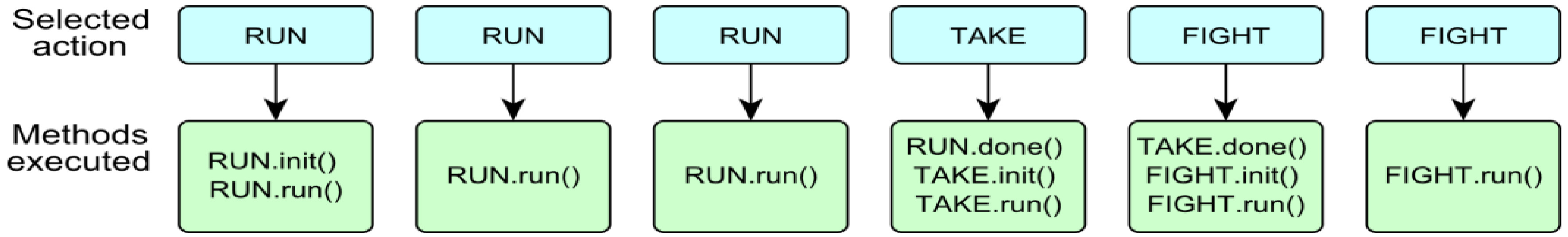
Created File: 2014 Pogamut VUT\27-yaPOSH-DuelBot\src\main\java\duelbot\sense\MyNewSense.java

3.) Edit generated Java source file

```
@PrimitiveInfo(name = "Can see a player", description = "Do I see a player?")
public class SeePlayer extends ParamsSense<AttackBotContext, Boolean> {
```

yaPOSH
Actions

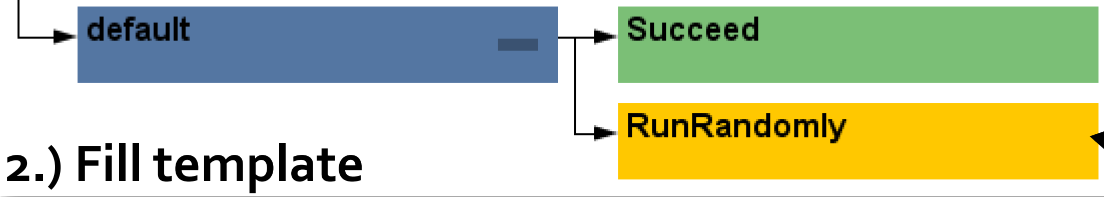
- Actions
  - Represent an action in the environment
  - Are expected to return:
    - **FINISHED** (an action has been finished successfully),
    - **RUNNING** (an IVA action is still being executed within the environment),
    - **FAILED** (an action execution has failed).
  - Have three methods – **init(), running(), done()**

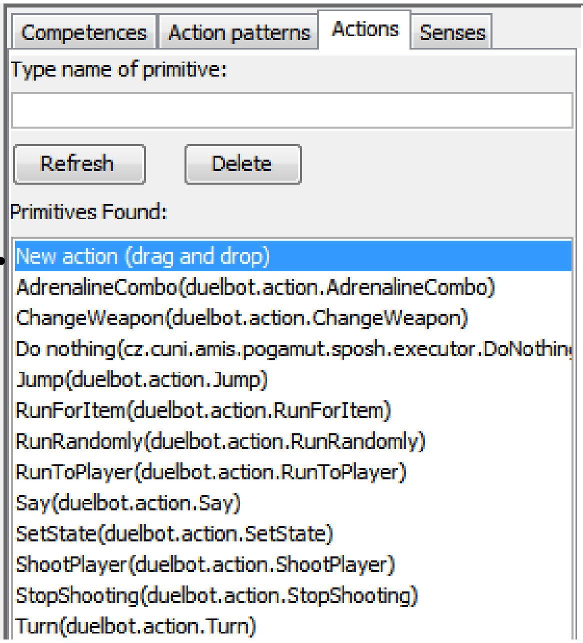


yaPOSH

How to make new Action?

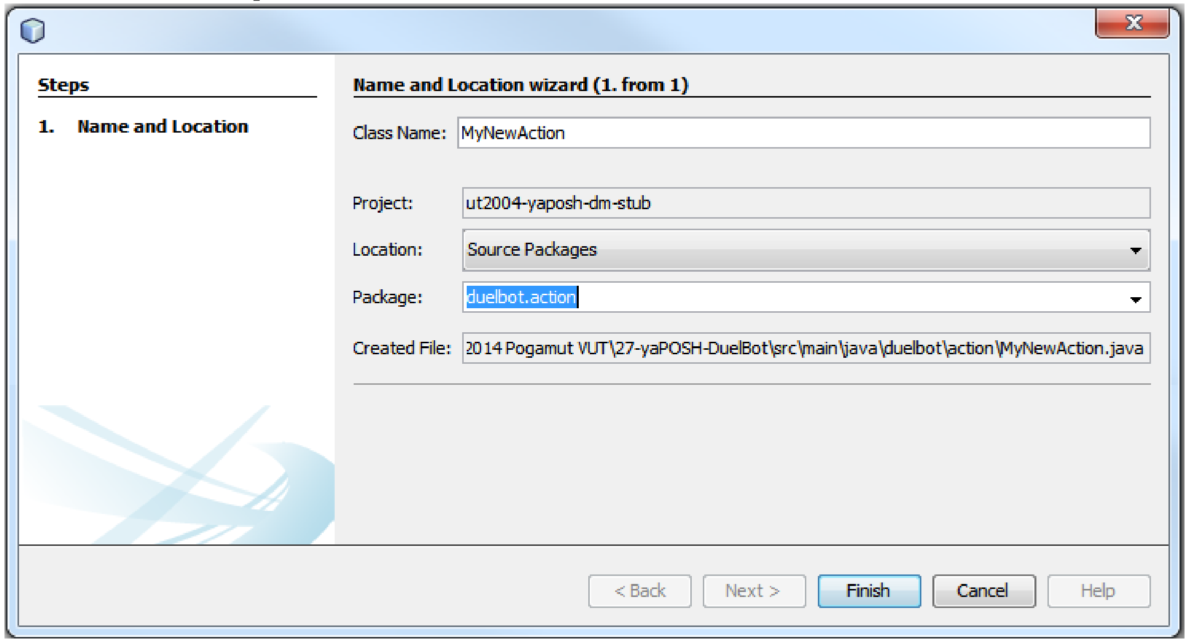
1.) DRAG & DROP!





New action (drag and drop)
AdrenalineCombo(duelbot.action.AdrenalineCombo)
ChangeWeapon(duelbot.action.ChangeWeapon)
Do nothing(cz.cuni.amis.pogamut.sposh.executor.DoNothin
Jump(duelbot.action.Jump)
RunForItem(duelbot.action.RunForItem)
RunRandomly(duelbot.action.RunRandomly)
RunToPlayer(duelbot.action.RunToPlayer)
Say(duelbot.action.Say)
SetState(duelbot.action.SetState)
ShootPlayer(duelbot.action.ShootPlayer)
StopShooting(duelbot.action.StopShooting)
Turn(duelbot.action.Turn)

2.) Fill template



Name and Location wizard (1. from 1)

Class Name:	MyNewAction
Project:	ut2004-yaposh-dm-stub
Location:	Source Packages
Package:	duelbot.action
Created File:	2014 Pogamut VUT\27-yaPOSH-DuelBot\src\main\java\duelbot\action\MyNewAction.java

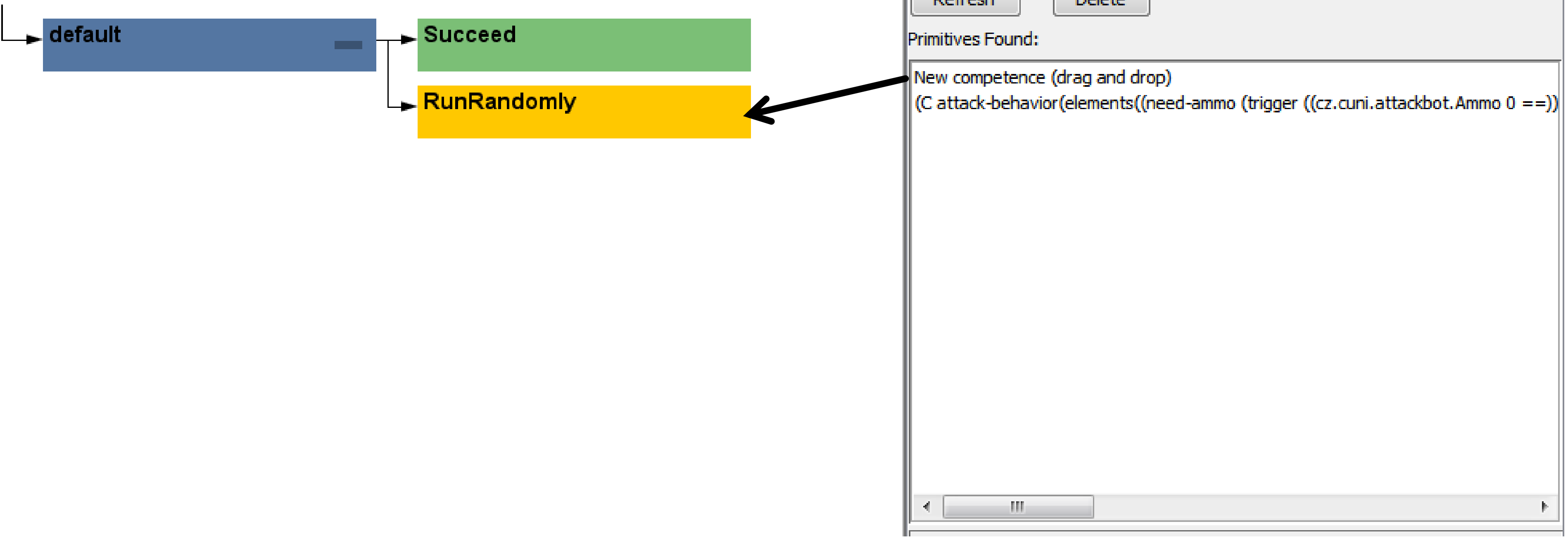
3.) Edit generated Java source file

```
@PrimitiveInfo(name="Shoot the player", description="Shoot the player.")
public class ShootPlayer extends ParamsAction<AttackBotContext> {
```

yaPOSH

New Action Pattern, New competence?

- Are created by drag and dropping from POSH editor from the tabs at the right side of IDE



New competence (drag and drop)
(C attack-behavior(elements((need-ammo (trigger ((cz.cuni.attackbot.Ammo 0 ==))

yaPOSH Context

How to access Pogamut modules?

- Every POSH action and sense has *Context* (this.ctx) that contains all Pogamut modules.
- *Context* is an editable class that is a part of your POSH bot sources, e.g. **AttackBotContext**
- You may use context to store some variables, e.g. *Item* you are going for or *Player* you are going to fight

yaPOSH

Parameters

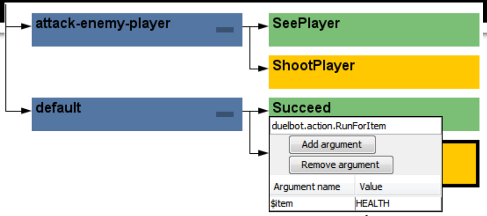
- Competences, action patterns, actions and senses can be parameterized

```
(
 (AP go-to-flag
  vars($target="enemy")
   (bot.TurnToFlag($teamname=$target)
    bot.GoToFlag($team=$target)
   )
 )

 (DC life
  (drives
   (
    (pickup-our-flag
     (trigger
      (
       (bot.FlagState($teamname="our")
                     "dropped")
       (bot.FlagIsVisible($teamname="our"))
     ))
     go-to-flag($target="our")
    )
   )
  )
 )
)
```

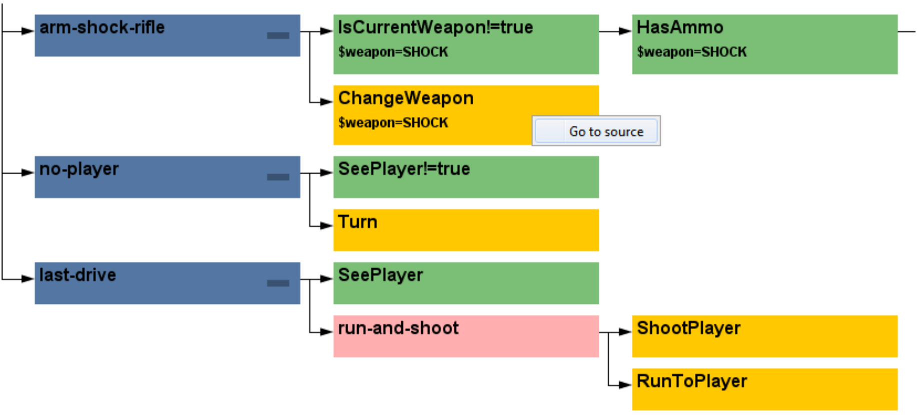
```
@PrimitiveInfo(name          = "Is flag visible",
               description = "our / enemy")
public class FlagVisible
       extends FlagSense<AttackBotContext,Boolean>
{
    public Boolean query(
           @Param("$teamname") String teamname
    ) {
        FlagInfo flag = getFlagInfo(teamname);
        return flag.isVisible();
    }

@PrimitiveInfo(name          = "Turn to flag",
               description = "our / enemy")
public class TurnToFlag
       extends FlagAction<AttackBotContext> {

    public ActionResult run(
           @Param("$teamname") String teamName
    ) {
        FlagInfo flag = getFlagInfo(teamName);
        ctx.getMove().turnTo(flag.getLocation());
        return ActionResult.RUNNING_ONCE;
    }
```

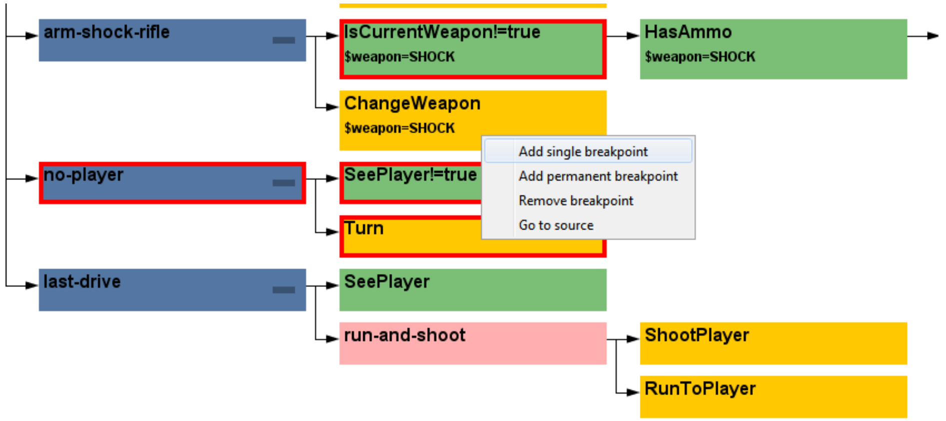
yaPOSH

POSH Editor



- Enables drag and drop
  - Select action or sense you want to add or change from the editor and drag and drop it at desired place
- Double clicking POSH graphical element opens editor, right clicking opens element menu
- Support “Go to source”, breakpoints and debugging
- Breakpoints **PAUSE** the bot **<u>AND</u>** the **environment**

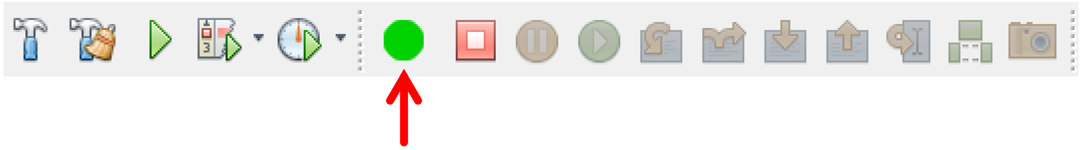




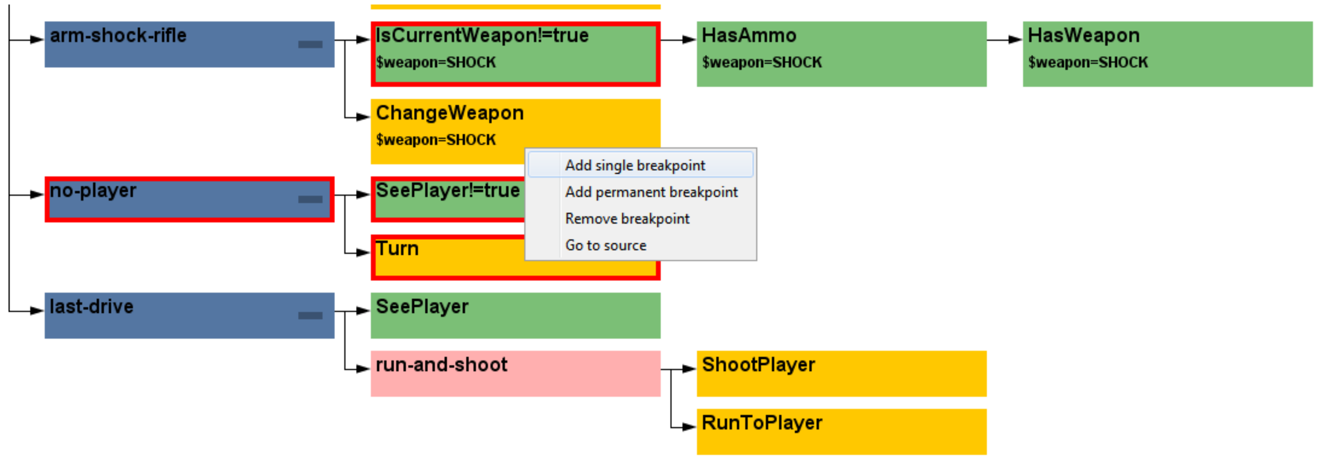
yaPOSH

How to run POSH plan debugger

- Run the bot in **Debug mode** (right click the project, select **Debug**)
- In the Debug toolbar, click the *green circle* button to enable POSH plan debugger



- A window with Debugger appears:



Practice Lesson

Outline

1. Big Picture
2. BOD (Behavior Oriented Design)
3. Gentle POSH introduction
4. **DeathMatch Bot**

Deathmatch Bot

Basics

- Its all about movement on the map
  - Picking the right place to be at
  - Picking the right item to go for
- Knowing when it is worth to change the behavior
  - I am almost at the rocket launcher, but I see enemy player. Will I go for the weapon or start fighting with the player?

Deathmatch Bot

Combat

- Using proper weapon in proper situations
  - `this.weaponPrefs` …
- Knowing how to move in combat
  - Strafing, dodging, jumping
  - Maintaining distance according bot current weapon
  - Facing one direction and move elsewhere (`navigation.setFocus(…)`)
- Beware that jumping and dodging reduces bot accuracy!

Assignment 8
(or Homework)

- Create **DeathMatchBot** in POSH
  - That arms himself and is able to fight an opponent
  - Does not stuck (for long).

DM Bot Tournament

Announcement!

- All your **DeathMatchBots** in yaPOSH will automatically take part in DM Bot Tournament
  - $1_{vs}1$, 10 frags, 10 minutes max, on one of $1_{vs}1$ dueling maps
- Deadline for the bots is **10.5.2014 23:59**
- Don't forget to send your bots to Jakub Gemrot as well even if you attend Monday lectures!

Assignment

Cheatsheet

- Access Pogamut modules from POSH actions and senses!
  - `this.ctx.getItems().getSpawnedItems(UT2004ItemType.Category.WEAPON)`
  - `MyCollections.getFiltered(Collection, new IFilter<Item>() {…})`
- Handling unreachable items:
  - `this.ctx.getNavigation().addStrongNavigationListener(…STUCK_EVENT…)`
  - `myTabooSet.add() & myTabooSet.filter(…)`
- Specifying weapon preferences:

  -
    ```
    this.ctx.getWeaponPrefs().addGeneralPref(UT2004ItemType.FLAK_CANNON,true)
      .addGeneralPref(UT2004ItemType.ROCKET_LAUNCHER,true);
    ```

Questions?

I sense a soul in search of answers…

- We do not own the patent of perfection (yet…)
- In case of doubts about the assignment, tournament or hard problems, bugs don't hesitate to contact us!
  - Jakub Gemrot (Tuesday practice lessons)
    - jakub.gemrot@gmail.com
  - Michal Bída (Monday practice lessons)
    - michal.bida@gmail.com