Faculty of Mathematics and Physics
Charles University in Prague
7th April 2015

UT2004 bots made easy!

Pogamut 3

Lecture 7 – Items and Weapons

Warm Up!

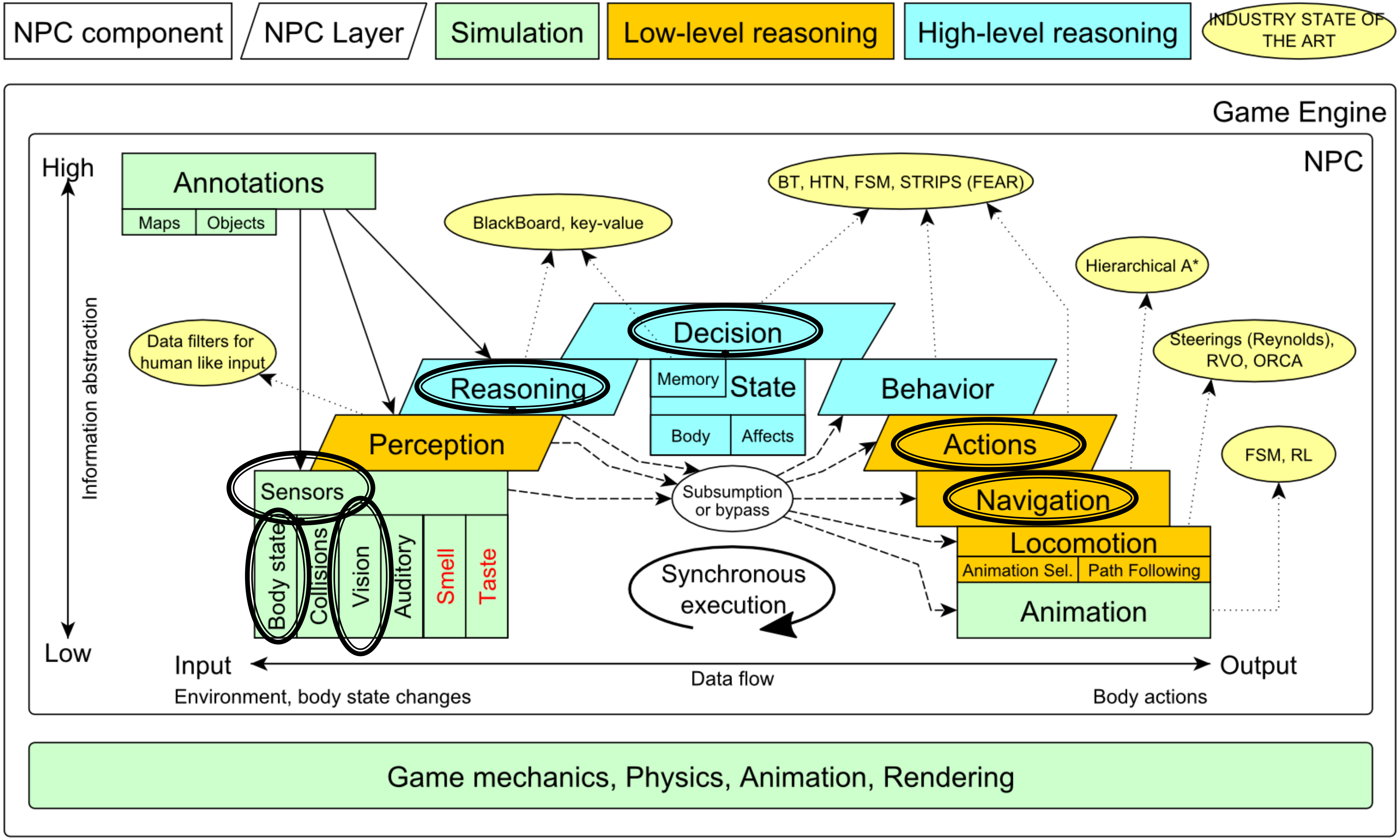
- Fill the short test for this lessons
  - 7 minutes limit
  - xxx
  - xxx

Today's menu

1. **Big Picture**
2. Pogamut World Abstraction
3. Navigation intermezzo
4. Items
5. Weapons & Shooting

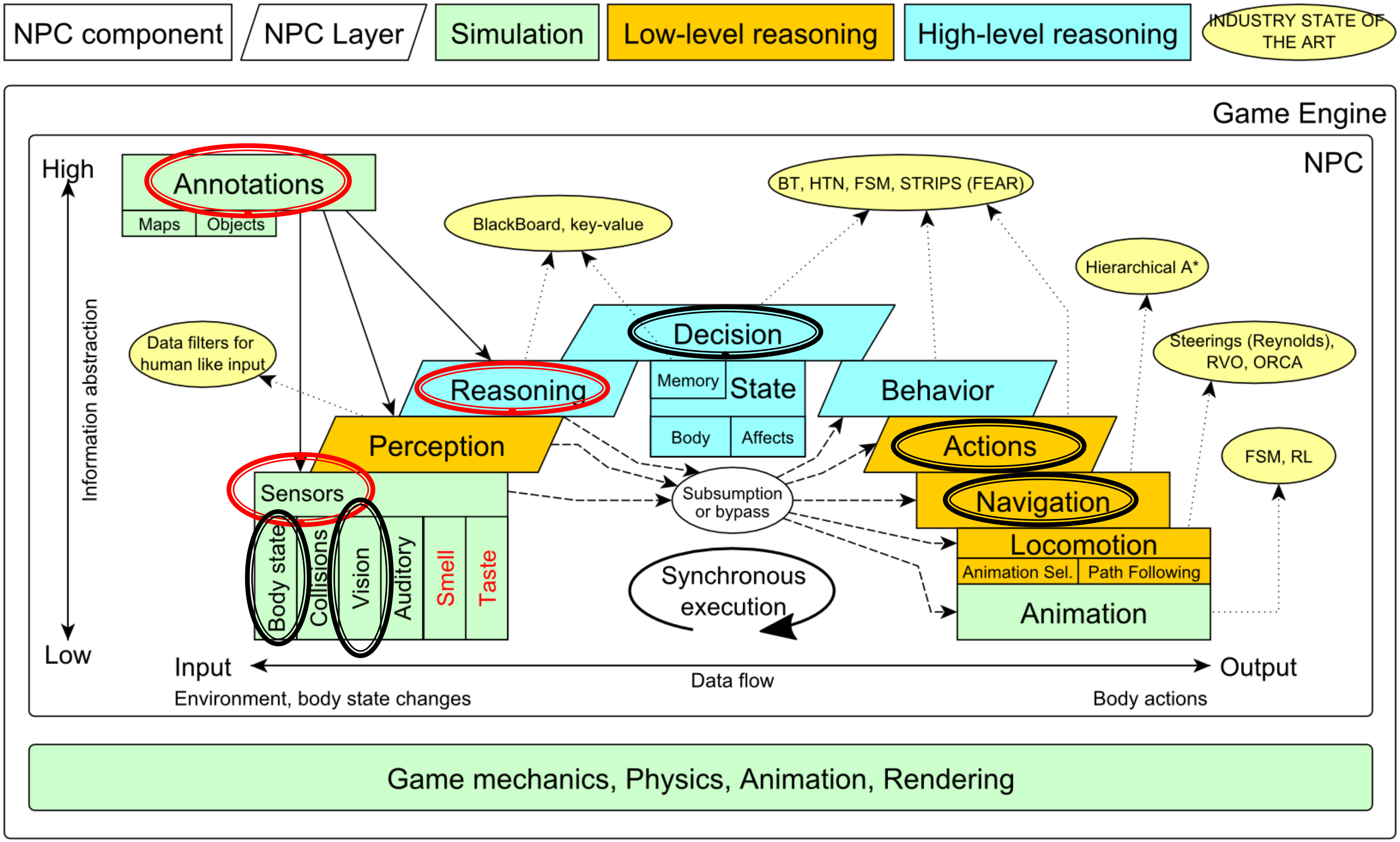
Big Picture

Already covered



Big Picture

Today



Today's menu

1. Big Picture
2. **Pogamut World Abstraction**
3. Navigation intermezzo
4. Items
5. Weapons & Shooting

Pogamut World Abstraction

Items overview

Objects (IWorldObject):

- Player
- **Item**
- NavPoint
- Self
- IncomingProjectile

Events (IWorldEvent):

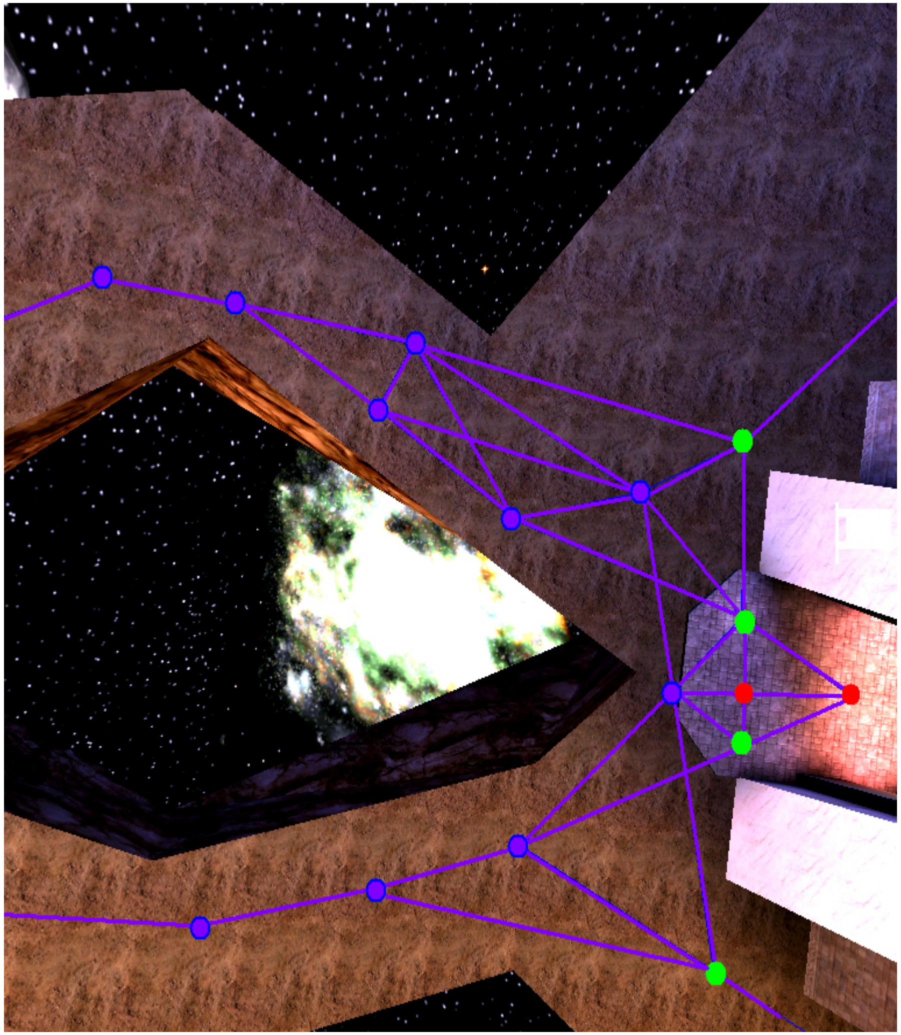
- HearNoise & HearPickup
- BotDamaged & BotKilled
- PlayerDamaged & PlayerKilled,
- **ItemPickedUp**
- GlobalChat

- Use modules, listeners and Pogamut helper classes!
  - this.players, **this.items**, this.info …
  - MyCollections, DistanceUtils, fwMap

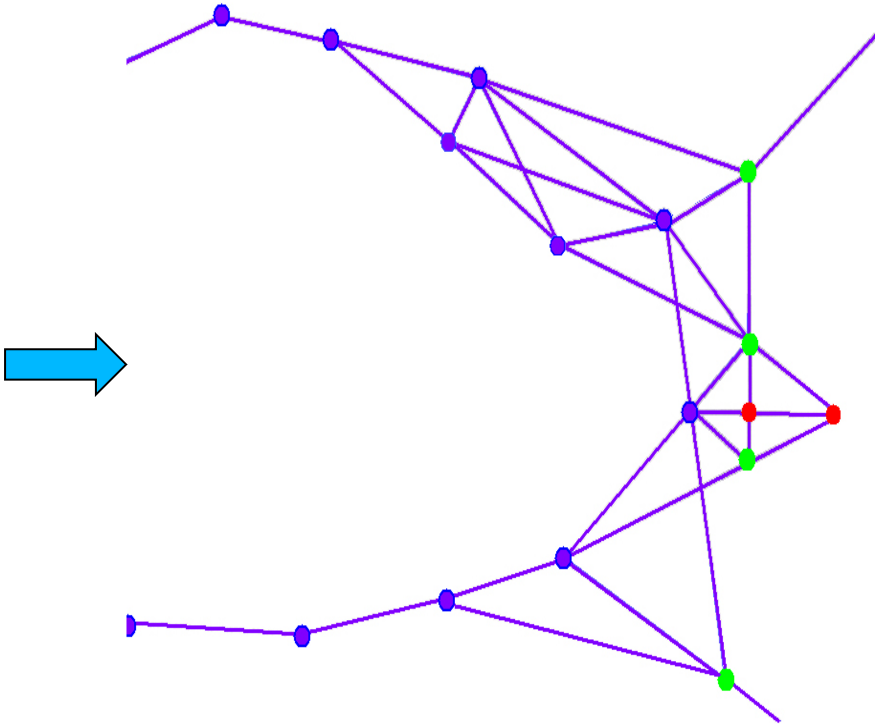
```
if (this.items. getSpawnedItems().values().size() > 0) {  …  }

@EventListener(eventClass = ItemPickedUp.class)
public void itemPickedUp(ItemPickedUp event) {
    …
}
```

UT2004 World Abstraction

Navigation graph



#Navpoints in the map = 100 – 5000



UT2004 World Abstraction

Nav link/NavPoint types

- **NavPoints**
  - **InventorySpot**
  - JumpPad
  - Lift
  - Teleport
  - Door
  - PlayerStart
  - SnipingSpot
  - …
- **Nav links**
  - Walk
  - Jump
  - Lift
  - Door
  - DoubleJump
  - …

Today’s menu

1. Big Picture
2. Pogamut World Abstraction
3. **Navigation intermezzo**
4. Items
5. Weapons & Shooting

Navigation

Step by step

General steps:

1. Decide where to go
2. Plan the path (list of navpoints/locations)
3. Follow the path

Navigation

Step by step

Story so far:

1. Decide where to go
2. Plan the path (list of navpoints)
3. Follow the path
    - **Watch for meaningfulness!**

4. **Check that you have truly grabbed the item!**

```
@EventListener(eventClass = ItemPickedUp.class)
public void itemPickedUp(ItemPickedUp event) {
    log.info("I've got an item! " +
    event.getType().getName());
}
```

Navigation

Stages

1. Decide where to go (Resoning + Decision making!)
   - `items.getSpawnedItems(UT2004ItemType)`
   - perform reasoning
     - It's OK to compute paths to all spawned items every `logic()`
2. Plan and follow the path
   - `navigation.navigate(item);`

Navigation

Stages

3. Follow the path
   - Do you still believe that item your running for is spawned?
     - It might have been picked up by your opponent!

```
if (!items.isPickupSpawned(item))
{ replan(); }
```

Navigation

Stages

4. Check that you truly grabbed the item!
   - UT2004 navigation is not 100% precise
     - It might stop running just right before the item!

```
@EventListener(eventClass = ItemPickedUp.class)
public void itemPickedUp(ItemPickedUp event) {
    if (itemRunning.getId().equals(event.getId()) {
        // I have picked the item!
    }
}
```

Navigation

PathBuilder

General steps:

1. Decide where to go
2. Plan the path (list of navpoints/locations)
   - Navigation mesh “problem”
   - Complex path-planning
   - `PathBuilder`
3. Follow the path

Navigation

NavMesh and PathBuilder

What's the difference between "navpoint path" and "navmesh path"?

- Navmesh path is truly the shortest
- Navmesh path does not traverse only nav-links
- Navmesh path likes to do "wall-hugging"

⇒ This means you mill miss moderate number of items you would have picked otherwise!

Navigation

NavMesh and PathBuilder

What is the real problem with “the shortest path to the item”?

You do not want that!

You need “a bit longer than shortest not the path to target item that picks other items along the way”.

Can you code this idea? …

Hardly…

Navigation

NavMesh and PathBuilder

What is the real problem with “the shortest path to the item”?

You do not want that!

You need “**a bit longer** than shortest not the path to target item that picks **other items** along the way”.

What is “a bit longer”?
What are “other items”?

Navigation

NavMesh and PathBuilder

What is the real problem with “the shortest path to the item”?

You do not want that!

You need “**a bit longer** than shortest not the path to target item that picks **other items** along the way”.

=> Price / Performance balancing

Navigation

NavMesh and PathBuilder

Solution? (Two approaches)

1. Plan the path in advance
   - Plan the shortest path
   - Search for detours
   - Refine as long as there are “interesting items nearby” and “detour is not long” (greedy way)
2. Look for opportunities along the way
   - Plan the shortest path
   - Start navigating
   - Look around if you cannot “replan the path”
   - If you still have a credit for that

Compare these approaches!

Today's menu

1. Big Picture
2. Pogamut World Abstraction
3. Navigation intermezzo
4. **Items**
5. Weapons & Shooting

Items

Basics

- `Item` (accesible via **`this.items`** !)
  - More “spawning location” than item
    - `items.`**`isPickupSpawned`**`(item)`
  - Unique `UnrealId` => Can be used in `Set`, `Map`
  - `ILocated` ~ `getLocation()` ~ X, Y, Z
  - `IViewable` ~ `isVisible()`
  - Always has corresponding `NavPoint` instance
    - `NavPoint itemNP = item.getNavPoint()`
  - Described by **`UT2004ItemType`**
    - `item.`**`getType()`**

Items

Important ItemTypes

```
UT2004ItemType.FLAK_CANNON
              .MINIGUN
              .LIGHTING_GUN
              .ROCKET_LAUNCHER
              .LINK_GUN

UT2004ItemType.SUPER_HEALTH
              .SUPER_ARMOR
              .SHIELD_PACK
              .SUPER_SHIELD_PACK
              .U_DAMAGE_PACK
```

Items

ItemType, UT2004ItemType & Categories

- `UT2004ItemType, ItemType`
  - Enum holding concrete type of the item
  - Part of some **`ItemType.Category`**
    - Categories are divided based on what items are intended to do
    - `ItemType.Category.`**`HEALTH`**
    - `ItemType.Category.`**`ARMOR`**
    - `ItemType.Category.`**`SHIELD`**
    - `ItemType.Category.`**`WEAPON`**
    - `ItemType.Category.`**`AMMO`**

Items

Items

- Agent module: `items`

  ```
  items.getAllItems()
  items.getVisibleItems(UT2004ItemType)
  items.getSpawnedItems(UT2004ItemType)
  items.isPickable(Item)
  ```

- `DistanceUtils`

  ```
  .getNearest(Collection<Ilocated>)
  .getNthNearest(n,Collection<Ilocated>)
  ```

- `fwMap`

  ```
  .getNearestItem(Collection<Item>)
  ```

Items

ItemDescriptor(s)

- Every item is “well” described

```
Item item =
    items.getAll(ItemType.Category.WEAPONS).values()
    .iterator().next();
WeaponDescriptor weaponDesc =
    (WeaponDescriptor)
    descriptors.getDescriptor(item.getType());
if (weaponDesc.getPriDamage() > 50) {
…
}
```

- `Ammo`/`Armor`/`HealthDescriptor` available as well

Today's menu

1. Big Picture
2. Pogamut World Abstraction
3. Navigation intermezzo
4. Items
5. **Weapons & Shooting**
   - http://planetunreal.gamespy.com/View.php?view=UT2004GameInfo.Detail&id=26

Weapons

UT2004 weapons guide I – the weak

- `UT2004ItemType.SHIELD_GUN (DEFAULT)`
  - Melee weapon (can be charged)
  - Secondary mode – shield (handy from distance!)
- `UT2004ItemType.ASSAULT_RIFLE (DEFAULT)`
  - Weak, basic, inaccurate (can have two)
  - Secondary mode – grenades (charged), don't use
- `UT2004ItemType.BIO_RIFLE`
  - Fires green blobs, short range, defense weapon
  - Secondary mode – charged (big blob)
  - Avoid … weak
- `UT2004ItemType.LINK_GUN`
  - Primary fires rather slow, but decent projectiles
  - Secondary – medium-to-short range beam
    - Max 2000 distance!

Weapons

UT2004 weapons guide II – the strong

- UT2004ItemType.FLAK_CANNON
  - Shotgun style weapon – deadly at short range
  - Sec. mode is a grenade launcher, don't use

- UT2004ItemType.MINIGUN
  - Choose between rapid fire but less accuracy (pri. mode) or slower fire and more accuracy (sec. mode)

- UT2004ItemType.SHOCK_RIFLE
  - Pri. mode is very accurate with medium damage
  - Sec. mode fires slow moving projectiles, that can be detonated by pri. fire making a big explosion (tricky to do though)

- UT2004ItemType.LIGHTING_GUN & UT2004ItemType.SNIPER_RIFLE
  - Sniper rifle – precise, can one-shot others by a headshot
  - Bots can use only pri. fire (sec. is zoom)

Weapons

UT2004 weapons guide III – mayhem

- `UT2004ItemType.ROCKET_LAUNCHER`
  - Good old rocket launcher, rockets have splash damage (beware!)
  - Secondary mode can charge up to three rockets

- `UT2004ItemType.REDEEMER`
  - Unleash nuclear mayhem!
    - big splash damage radius
  - Bots can use only primary firing mode!

- `UT2004ItemType.U_DAMAGE_PACK`
  - Not enough damage? Grab DOUBLE DAMAGE pack and double your damage output!

Weapons

Weaponry class

- `this.weaponry`
  - All you wanted to know about UT2004 weapons but were afraid to ask
  - Note that it contains also some obsolete and to-be-deprecated methods…

```
weaponry.getCurrentWeapon()
weaponry.hasWeapon(UT2004ItemType)
weaponry.hasLoadedWeapon()
weaponry.hasPrimaryLoadedWeapon()
weaponry.hasSecondaryLoadedWeapon()
weaponry.getLoadedWeapons()
weaponry.changeWeapon()
…
```

Weapons & Shooting

WeaponPreferences

- Weapons’ effectiveness depends on distance to target
- Thus you should create different priority list for various “ranges”
- Wrapped in class **`weaponPrefs`**

```
weaponPrefs.addGeneralPref(UT2004ItemType.MINIGUN,  true);
weaponPrefs.addGeneralPref(UT2004ItemType.LINK_GUN, false);
```

- true -> primary firing mode
- false -> secondary firing mode

```
weaponPrefs.newPrefsRange(CLOSE_COMBAT_RANGE = 300)
   .add(UT2004ItemType.FLAK_CANNON, true)
   .add(UT2004ItemType.LINK_GUN,    true); // 0-to-CLOSE
weaponPrefs.newPrefsRange(MEDIUM_COMBAT_RANGE = 1000)
   .add(UT2004ItemType.MINIGUN,             true)
   .add(UT2004ItemType.ROCKET_LAUNCHER,     true); // CLOSE-to-MEDIUM
```

- If range prefs fails, general are used
- You have to experiment! *(== behavior parametrization!)*

More at: **http://pogamut.cuni.cz/pogamut_files/latest/doc/tutorials/10-HunterBot.html**

Weapons & Shooting

Shooting

- Shooting with **`WeaponPrefs`** is easy!

```
Player enemy = players.getNearestVisiblePlayer();

shoot.shoot(weaponPrefs, enemy);

shoot.shoot(weaponPrefs, enemy,
    UT2004ItemType.ROCKET_LAUNCHER);
// do not use rocket launcher

shoot.setChangeWeaponCooldown(millis);
```

Weapons & Shooting

Time your shooting – Cooldown class

- Sometimes you need to perform the behavior “once in a time” => Cooldown

```
Cooldown rocketCD = new Cooldown(2000);
                           // millis

if (rocketCD.isCool()) {
   rocketCD.use();
   shoot.shoot(weaponPrefs, enemy);
} else {
   shoot.shoot(weaponPrefs, enemy,
   UT2004ItemType.ROCKET_LAUNCHER);
}
```

Weapons & Shooting

Time your behaviors – Heatup class

- Sometimes you need to pursue some behavior for a while => **Heatup**

```
Heatup pursueEnemy = new Heatup(3000);
                           // millis

if (players.canSeeEnemy()) {
   pursueEnemy.heat();
   // fight the enemy
} else
if (pursueEnemy.isHot()) {
   // pursue the enemy
} else {
   // collect items
}
```

Assignment 7

(or Homework)

- Create **CollectorBot**
  - Collects weapons, ammo and armor on the map
  - Run 3 bots on ***DM-1on1-Albatross***
  - What if the item you want to pick up is not there? (e.g. you run two collector bots and the other one got it first) ~ items**.isPickupSpawned(**item**)**
    - Re-plan!
  - How to check that your bot can pick some item?
    - `items.isPickable(item)`
  - How to check the bot successfully picked up an item?
  - How to avoid unreachable items?
    - Use **TabooSet**

Assignment

Cheatsheet

- Getting and filtering the items:
  - `this.items.getSpawnedItems(UT2004ItemType.Category.WEAPON)`
  - `MyCollections.getFiltered(Collection, new IFilter<Item>() {…})`
- Handling unreachable items:
  - `Navigation.addStrongNavigationListener(…STUCK_EVENT…)`
  - `myTabooSet.add() & myTabooSet.filter(…)`
- Some thin items (e.g. *HealthVial*) are tricky to pick up! How to be sure that your bot has picked the item up?
  - `ItemPickedUp.class` event

    ```
    @EventListener(eventClass=ItemPickedUp.class)
    public void pickedUp(ItemPickepUp event) {}
    ```

Assignment

Cheatsheet

- How can I know that the item is pickable?
  - When bot's helath is 100, MEDKIT is not pickable…
  - `if (this.items.isPickable(item)) { … }`
    - `items.isPickable()` tells you whether you can pick the item up at all!

Send us finished assignment

Via e-mail:

- *Subject*
  - “Pogamut homework 2015 – Assignment X”
    - Replace `‘X’` with the assignment number and the subject has to be without quotes of course
    - …or face -2 score penalization
- *To*
  - jakub.gemrot@gmail.com
    - Jakub Gemrot (Tuesday practice lessons)
- *Attachment*
  - Completely zip-up your project(s) folder except `‘target’` directory and `IDE specific files` (or face -2 score penalization)
- *Body*
  - **Please send us information about how much time it took you to finish the assignment + any comments regarding your implementation struggle**
    - *Information won’t be abused/made public*
    - *In fact it helps to make the practice lessons better*
  - Don’t forget to mention your full name!

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