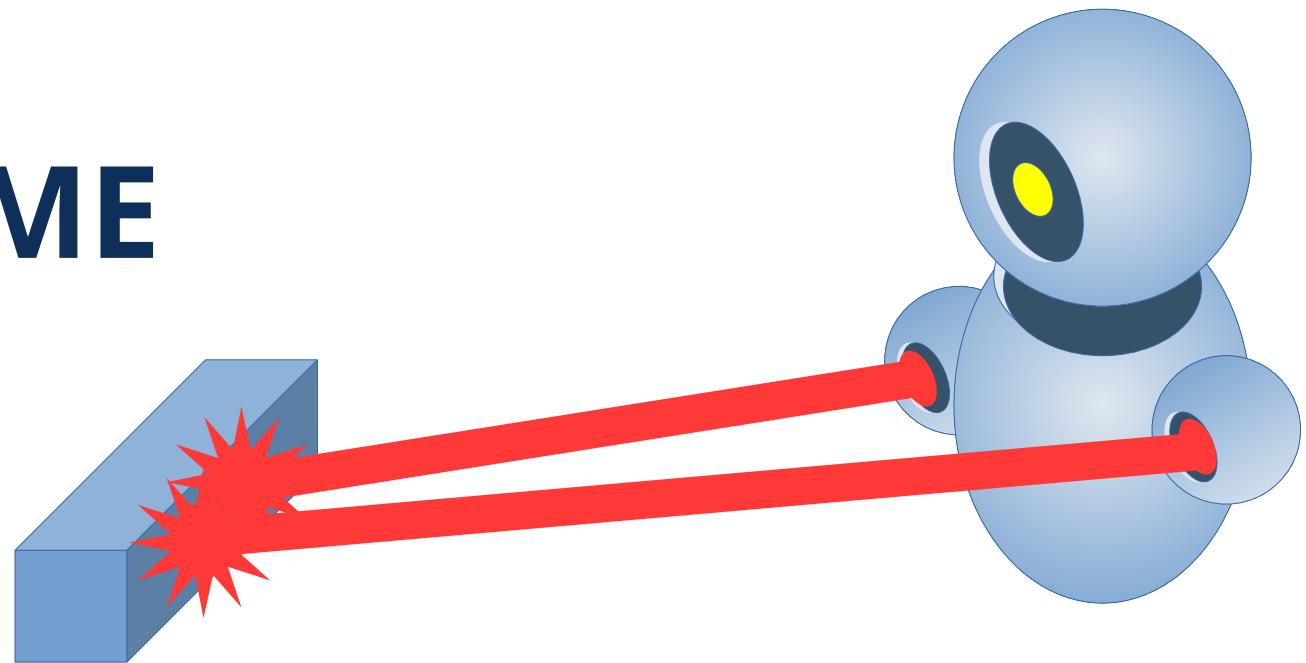
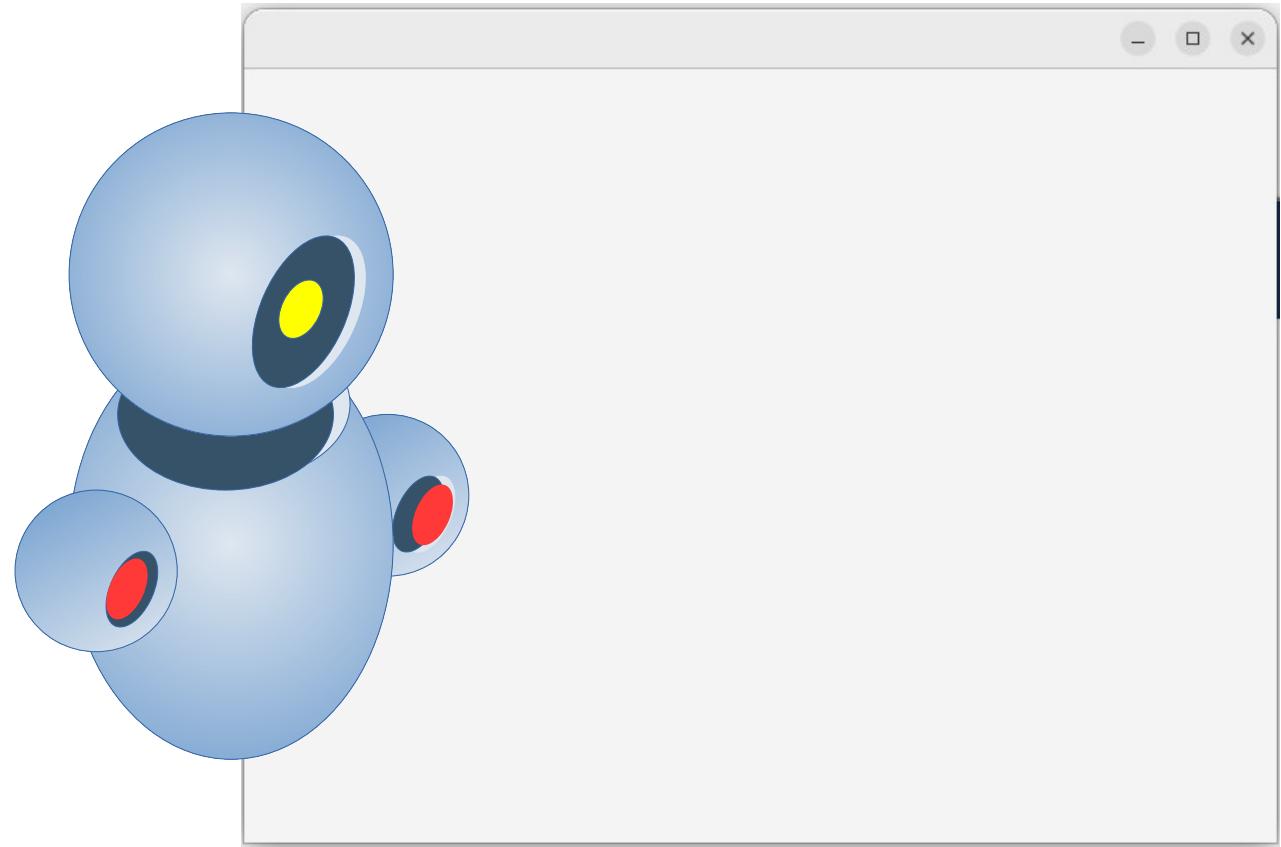


# THE CHON GAME IN JAVA

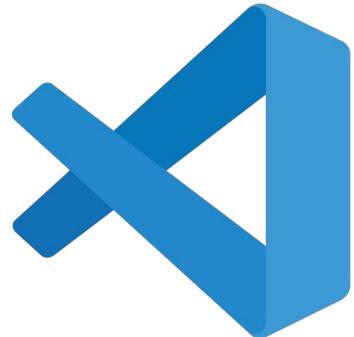


# MY FIRST JAVA FX APPLICATION

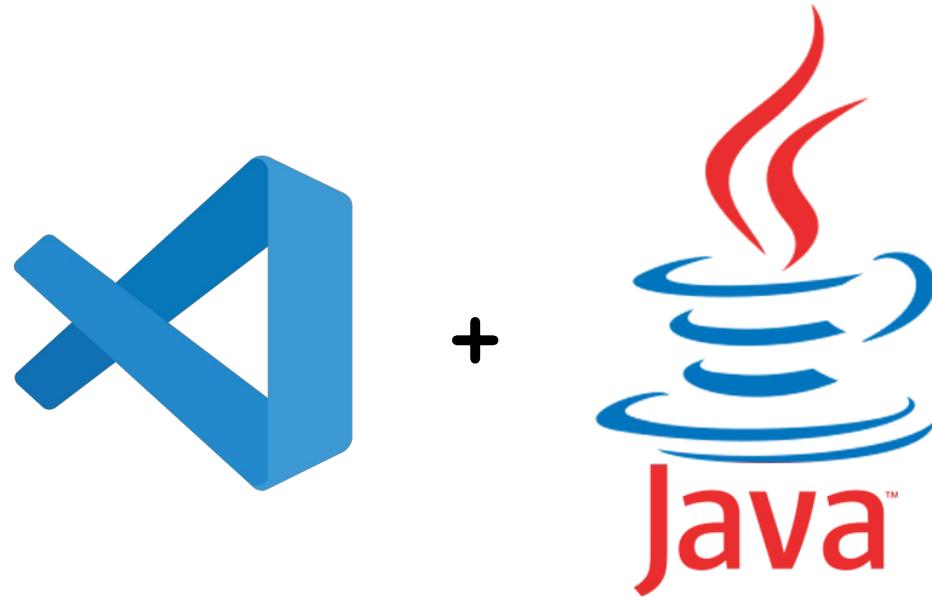


JavaFX is a comprehensive software platform for creating and programming client and cross-platform desktop applications.

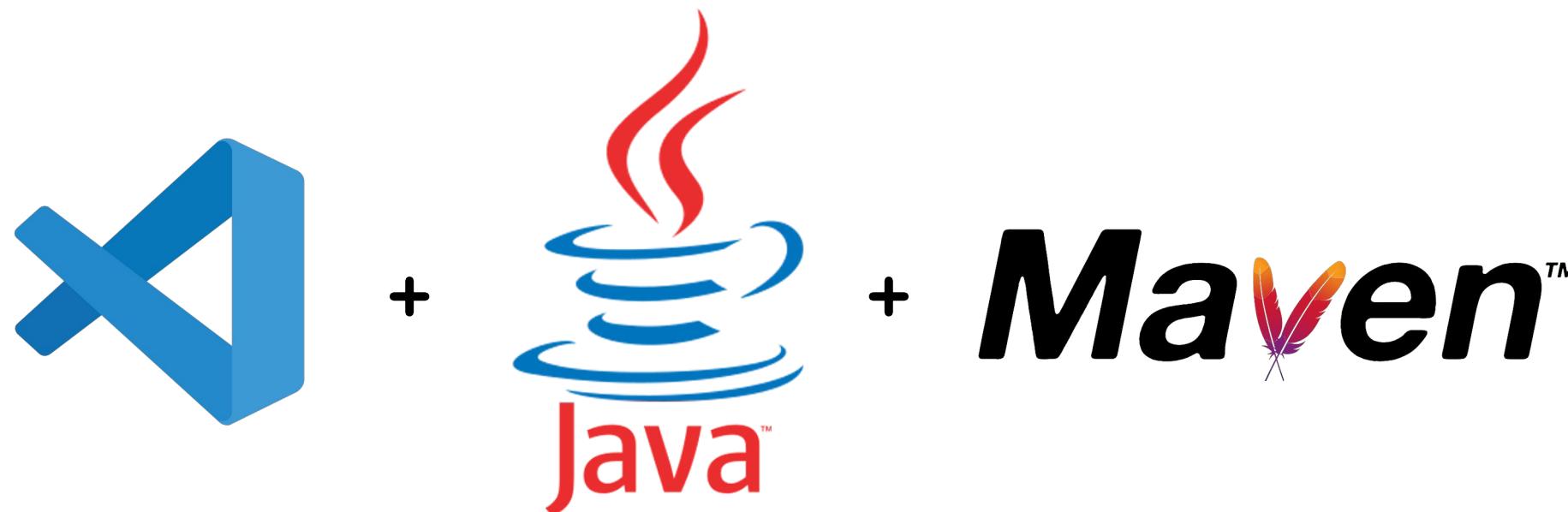
# Project Technologies



# Project Technologies

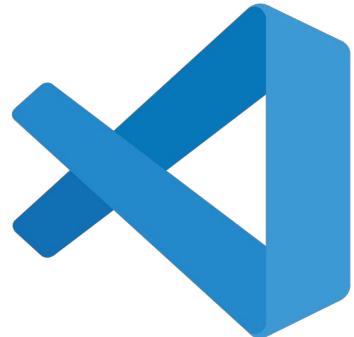


# Project Technologies

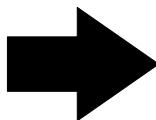
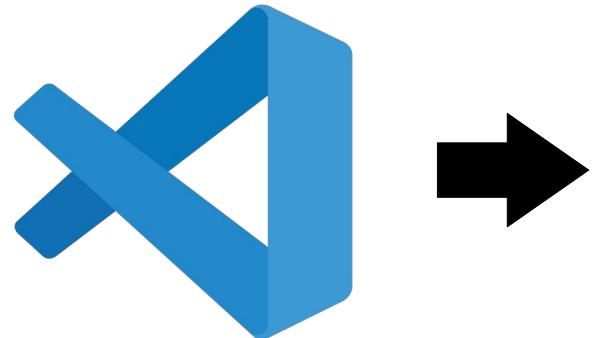


# Project Technologies





# VSCode



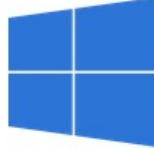
https://code.visualstudio.com/download

Visual Studio Code Docs Updates Blog API Extensions FAQ Learn

Version 1.87 is now available! Read about the new features and fixes from February.

## Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.

 Windows  
Windows 10, 11

 .deb  
Debian, Ubuntu

 .rpm  
Red Hat, Fedora, SUSE

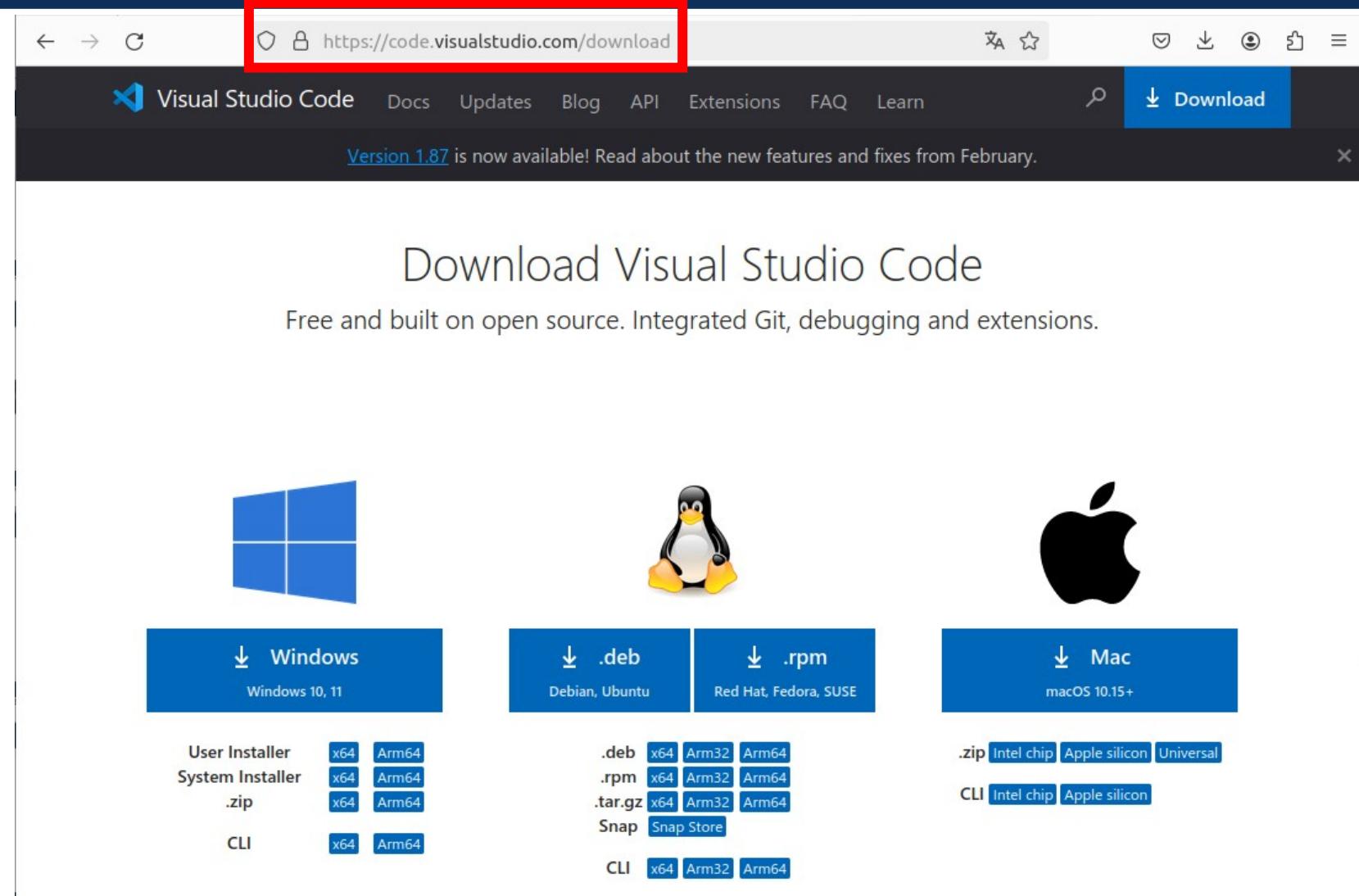
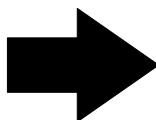
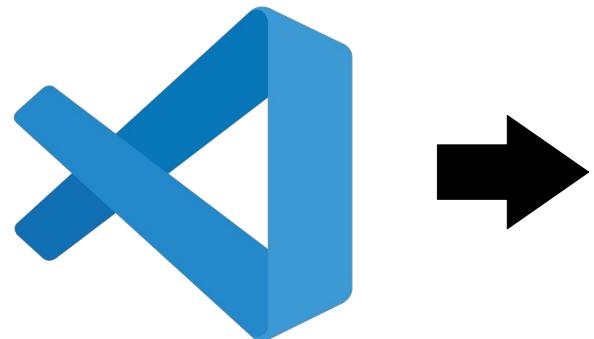
 Mac  
macOS 10.15+

User Installer  
System Installer  
.zip  
CLI

.deb  
.rpm  
.tar.gz  
Snap  
CLI

x64  
Arm64  
x64  
Arm64  
x64  
Arm64  
x64  
Arm32  
Arm64  
x64  
Arm32  
Arm64  
Snap Store  
x64  
Arm32  
Arm64

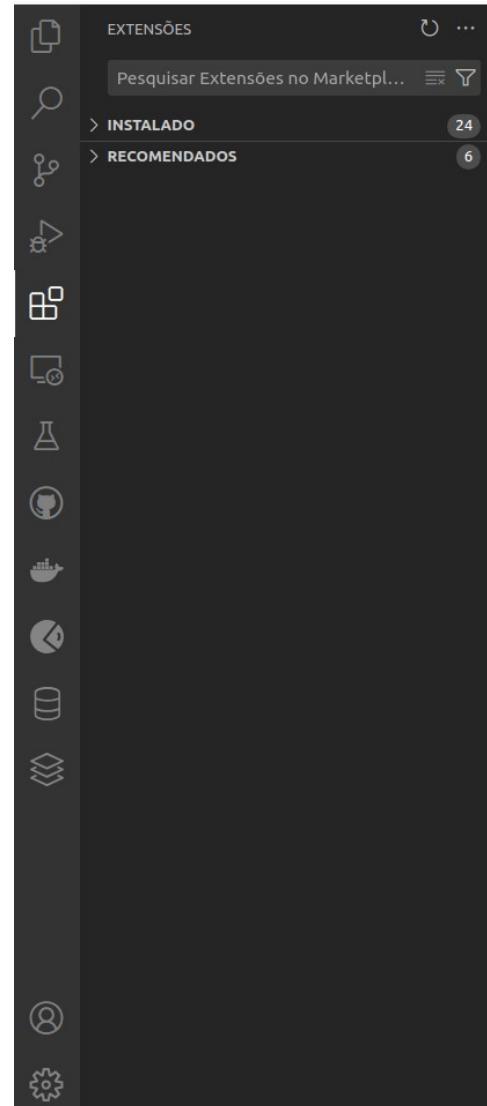
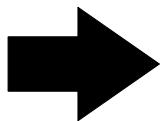
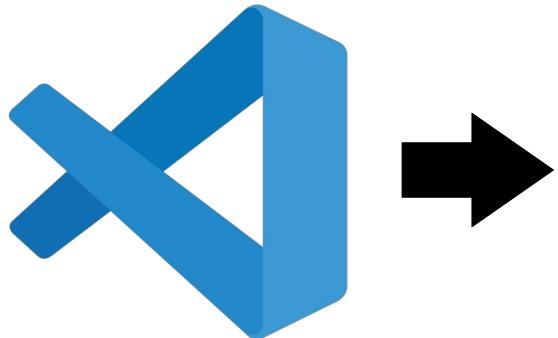
# VSCode



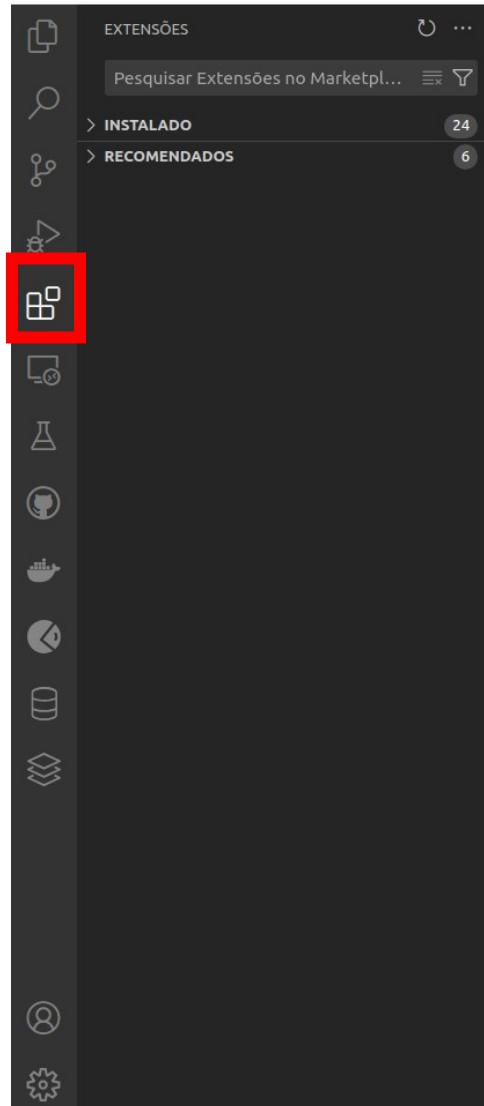
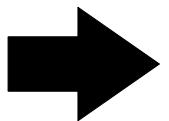
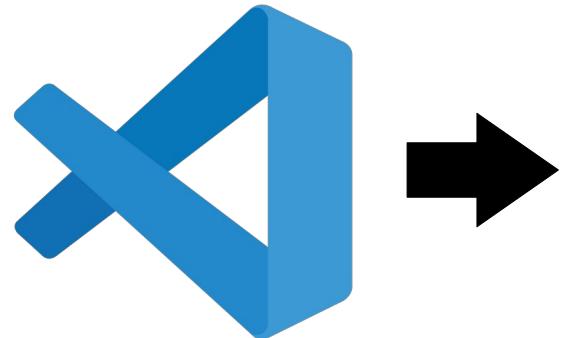
The screenshot shows the official Visual Studio Code download page at <https://code.visualstudio.com/download>. The page features a dark header with the Visual Studio Code logo and navigation links for Docs, Updates, Blog, API, Extensions, FAQ, and Learn. A prominent blue 'Download' button is located in the top right. A banner at the top of the main content area announces 'Version 1.87 is now available! Read about the new features and fixes from February.' Below this, the title 'Download Visual Studio Code' is displayed, followed by the subtitle 'Free and built on open source. Integrated Git, debugging and extensions.' The page then lists download links for three major operating systems:

- Windows**: Available for Windows 10, 11. Options include User Installer (x64, Arm64), System Installer (x64, Arm64), .zip (x64, Arm64), and CLI (x64, Arm64).
- .deb**: Available for Debian, Ubuntu. Options include .deb (x64, Arm32, Arm64), .rpm (x64, Arm32, Arm64), .tar.gz (x64, Arm32, Arm64), and Snap (Snap Store).
- .rpm**: Available for Red Hat, Fedora, SUSE. Options include .deb (x64, Arm32, Arm64), .rpm (x64, Arm32, Arm64), .tar.gz (x64, Arm32, Arm64), and Snap (Snap Store).
- Mac**: Available for macOS 10.15+. Options include .zip (Intel chip, Apple silicon, Universal), CLI (Intel chip, Apple silicon), and .tar.gz (Intel chip, Apple silicon).

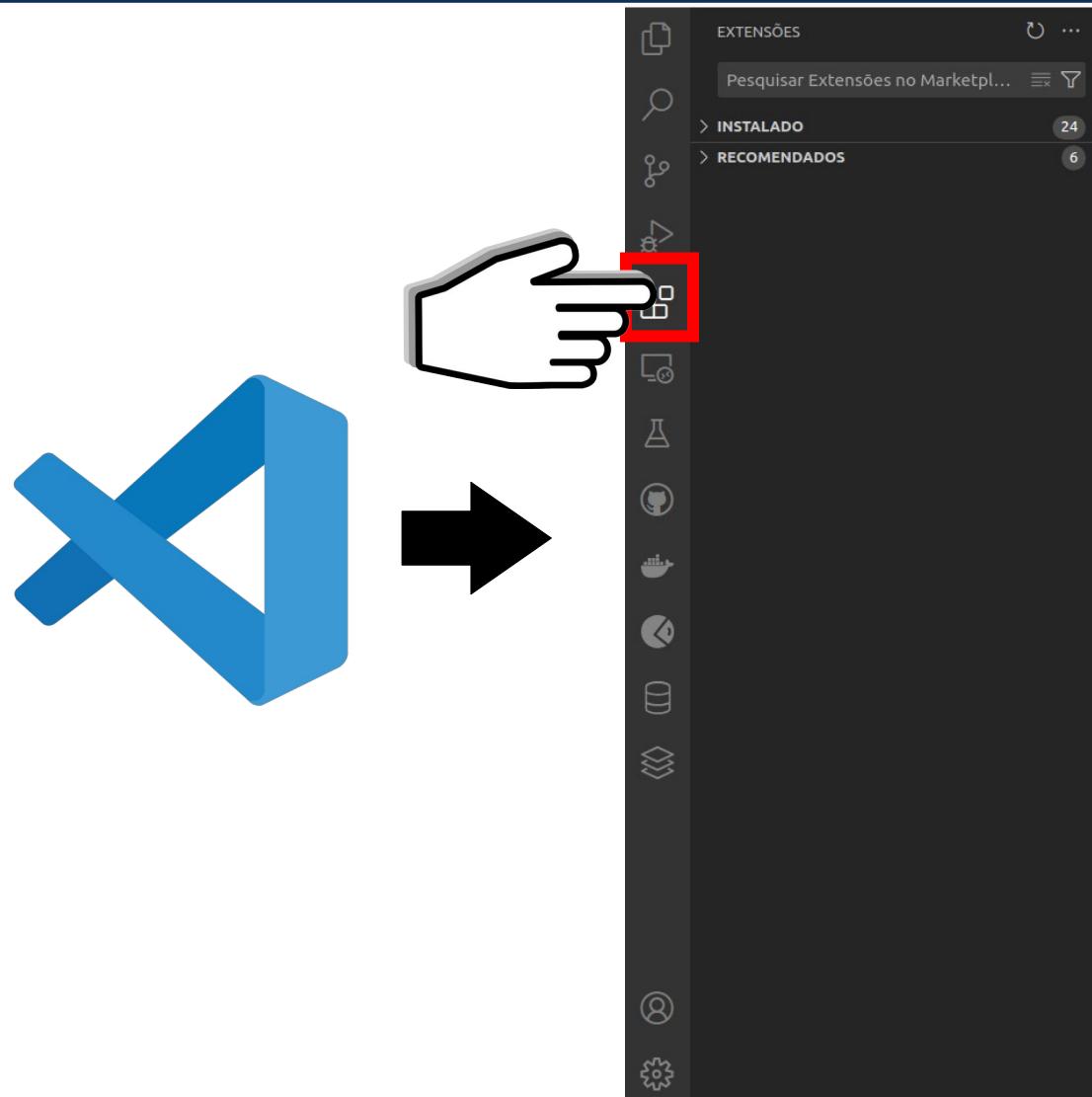
# VSCode



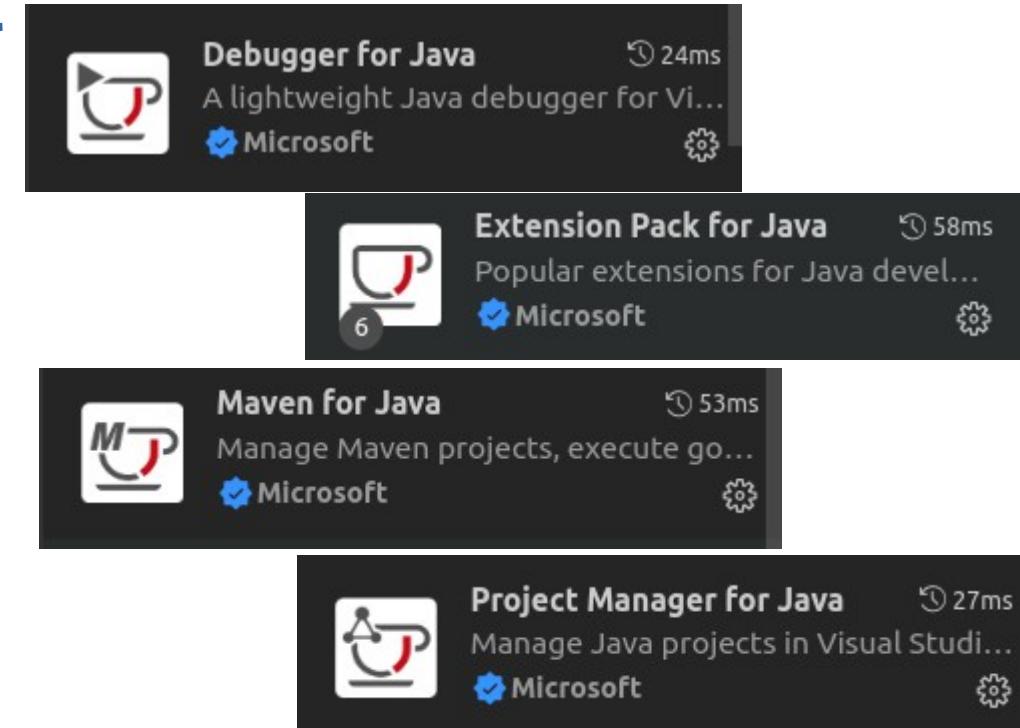
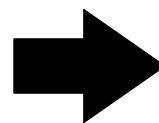
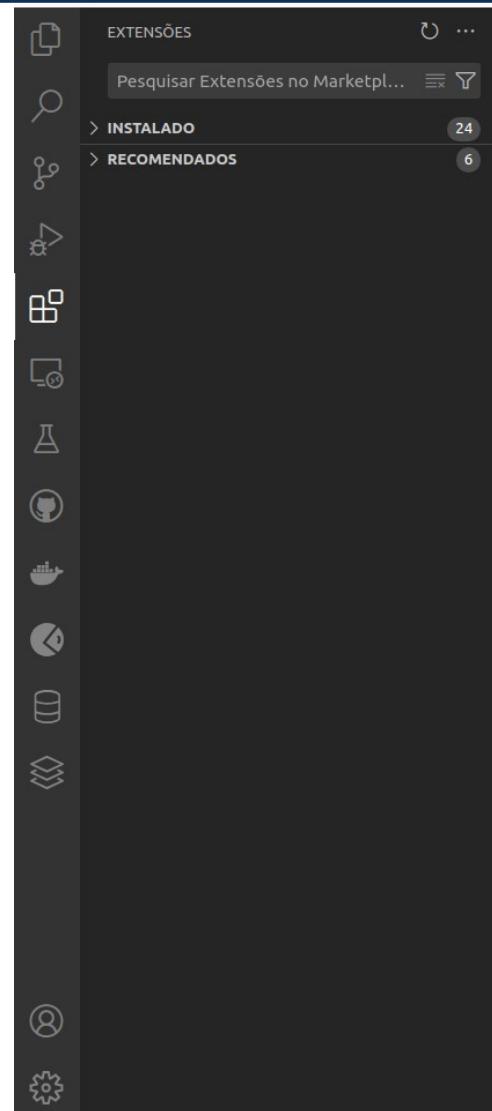
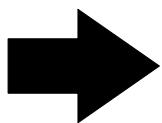
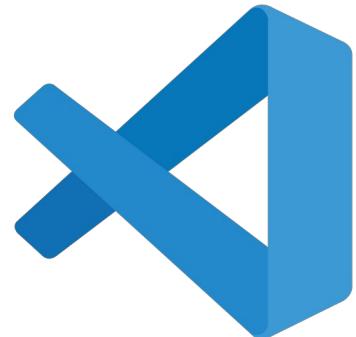
# VSCode



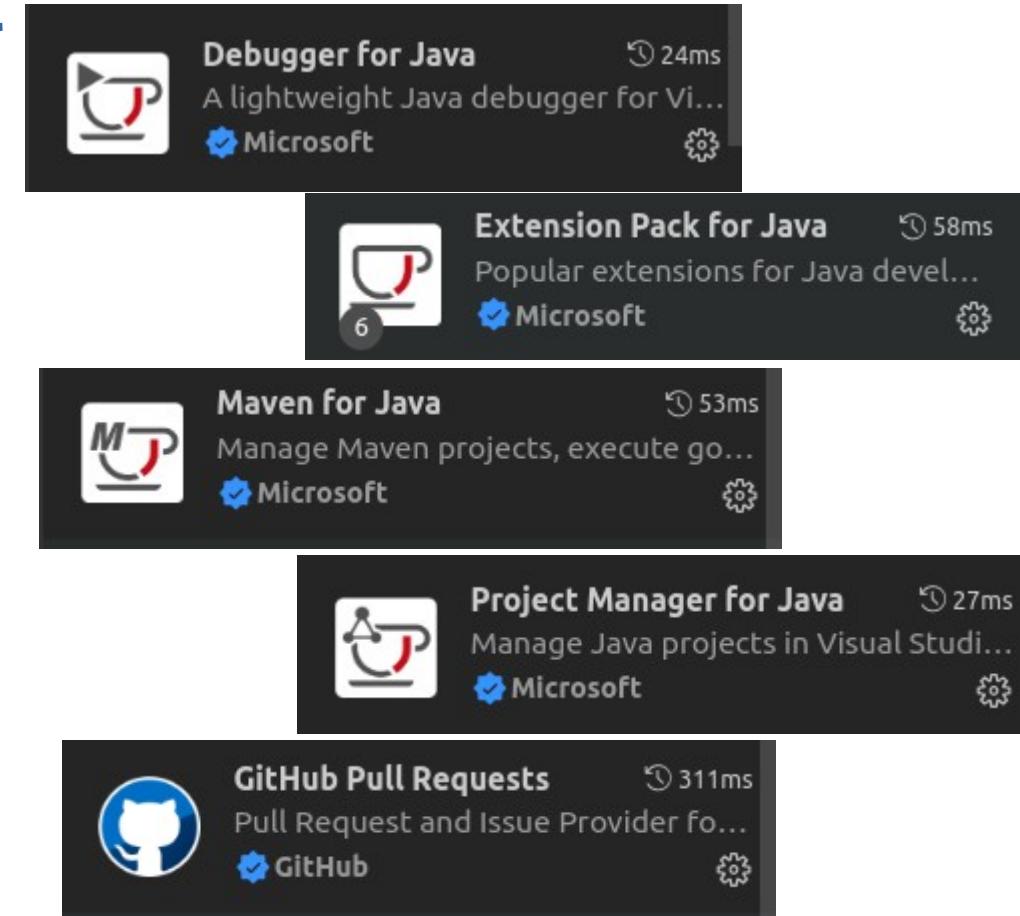
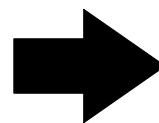
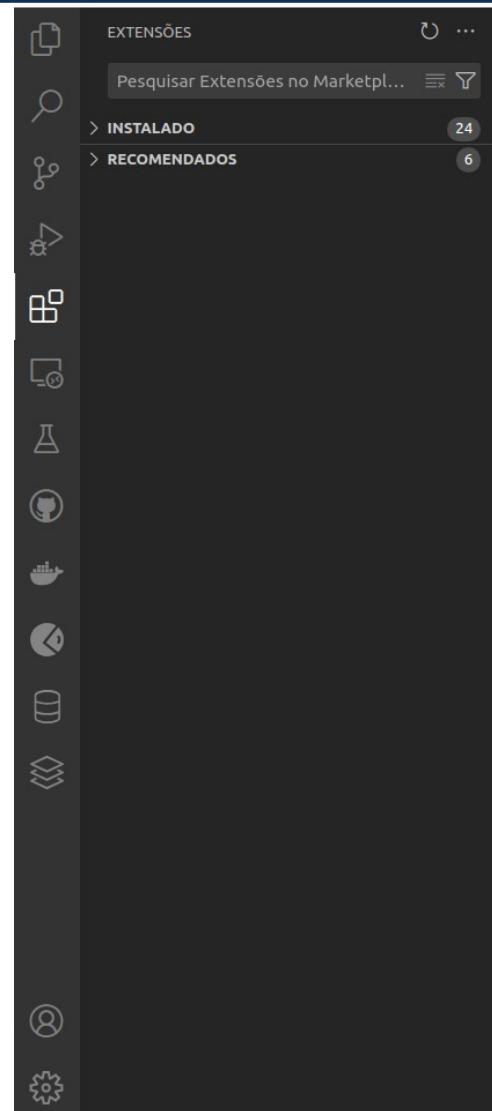
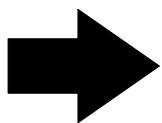
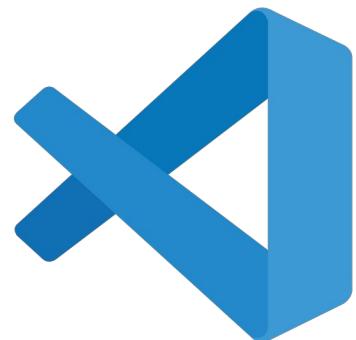
# VSCode



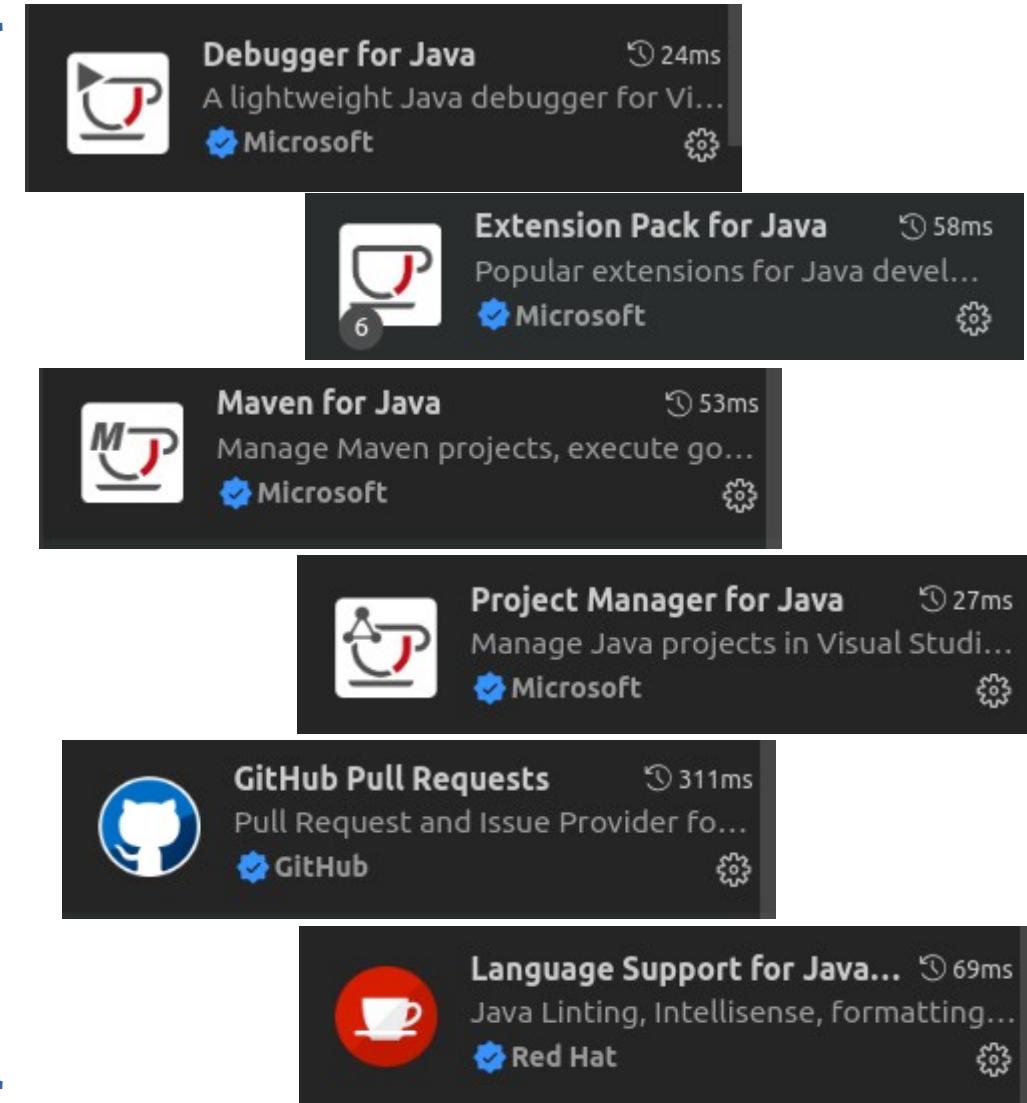
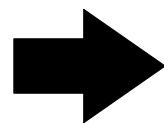
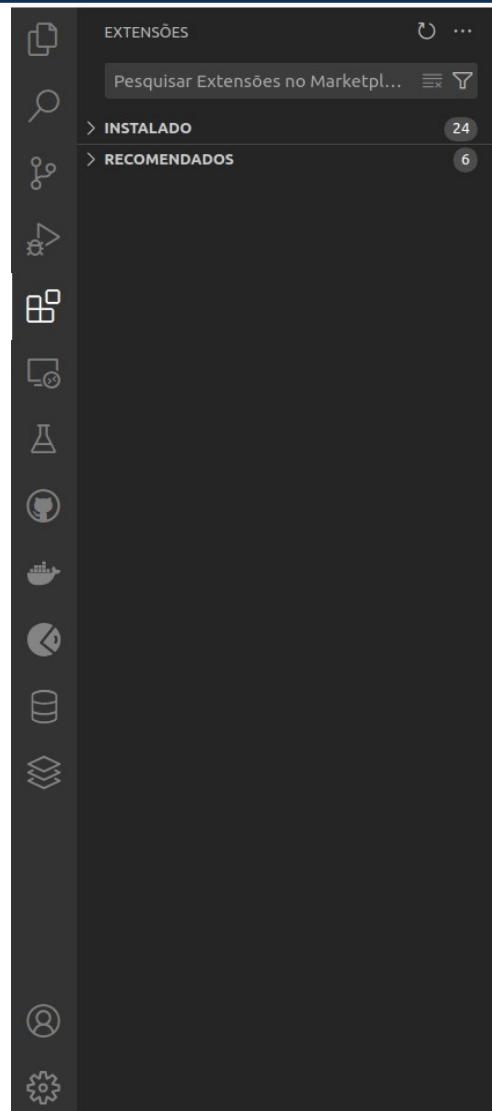
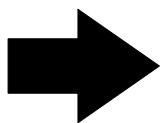
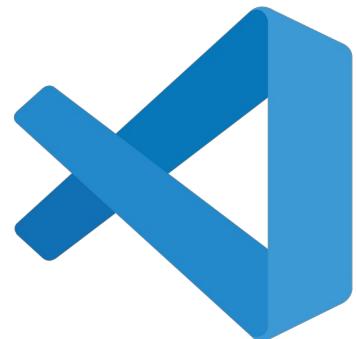
# VSCode



# VSCode

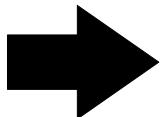


# VSCode





# GitHub



The screenshot shows a GitHub profile page for the user 'profpantoja'. At the top, there's a navigation bar with links for Overview, Repositories (10), Projects, Packages, Stars (2), and a search bar. Below the navigation is a circular profile picture of a man with rainbow hair, holding a whiteboard marker. The main content area starts with a bio section: 'Carpe diem. Make your lives extraordinary.' followed by a paragraph about the user's work in Artificial Intelligence, Robotics, Ubiquitous Computing, and the Internet of Things. It also lists current projects and collaborations. Below this is a 'Research Groups and Projects' section listing membership in CHON, OBINVEST, and the Turing Project. The 'Languages and frameworks I've interacted' section includes Java, MySQL, and other icons. At the bottom, there's a 'Follow me on Social Media' section with links to LinkedIn, ResearchGate, Google Scholar, iD, and Instagram.

profpantoja / README.md

**Carpe diem. Make your lives extraordinary.**

I work in Artificial Intelligence, more specifically in Multi-agent Systems and Software Engineering, but I have ventured into the areas of Robotics, Ubiquitous Computing, and the Internet of Things. In the management area, I work in technology dissemination (and currently social media), technological innovation, entrepreneurship, and Information Technology Governance and process mapping.

- I'm currently working on [multi-agent-system](#) [embedded-system](#) [modeling-language](#) [metamodel](#)
- I'm collaborating on [JaCaMo](#) [ChonIDE](#)

**Research Groups and Projects**

- Member of Cognitive Hardware on Networks Research Group (CHON).
- Co-coordinator of the Brazilian Investment Olympics (OBINVEST).
- Founder of the Turing Project.

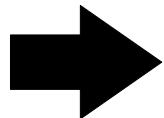
**Languages and frameworks I've interacted**

Java MySQL

**Follow me on Social Media**

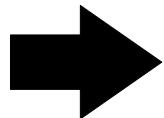
LinkedIn ResearchGate Google Scholar iD Instagram

# GitHub



The screenshot shows a GitHub profile page for the user 'profpantoja'. At the top, there's a navigation bar with tabs: 'Overview' (selected), 'Repositories' (highlighted with a red box), 'Projects', 'Packages', and 'Stars' (2). Below the navigation bar is a circular profile picture of a man with rainbow hair, holding a whiteboard marker. The profile section includes the name 'Kadu Pantoja', the GitHub handle 'profpantoja', and a brief bio: 'Professor and researcher at CEFET/RJ, master and Ph.D in Computing at the Military Institute of Engineering (IME) and UFF with mobility at Sorbonne UPMC.' It also shows 'Edit profile', '62 followers', and '8 following'. Below the profile is a 'Repositories' section, followed by sections for 'Research Groups and Projects', 'Languages and frameworks I've interacted', and 'Follow me on Social Media' with links to LinkedIn, ResearchGate, Google Scholar, iD, and Instagram.

# GitHub



The screenshot shows a GitHub profile page for the user 'profpantoja'. At the top, there's a navigation bar with tabs: 'Overview' (selected), 'Repositories' (highlighted with a red box and a cursor hand icon), 'Projects', 'Packages', and 'Stars 2'. Below the navigation is a circular profile picture of a man with rainbow hair, holding a whiteboard marker. The profile section includes the name 'Kadu Pantoja', the handle 'profpantoja', a bio about working in Artificial Intelligence and other fields, and a list of current projects and collaborations. The main content area features sections for 'Research Groups and Projects' (listing CHON, OBINVEST, and the Turing Project), 'Languages and frameworks I've interacted' (listing Java, MySQL, and others), and 'Follow me on Social Media' (with links to LinkedIn, ResearchGate, Google Scholar, iD, and Instagram).

https://github.com/profpantoja

profpantoja

Overview Repositories Projects Packages Stars 2

Kadu Pantoja  
profpantoja

Professor and researcher at CEFET/RJ, master and Ph.D in Computing at the Military Institute of Engineering (IME) and UFF with mobility at Sorbonne UPMC.

Edit profile

62 followers · 8 following

CEFET/RJ  
Rio de Janeiro  
<https://turing.pro.br/kadupantoja>  
[prof.pantoja](mailto:prof.pantoja)

Type ⌘ to search

profpantoja / README.md

**Carpe diem. Make your lives extraordinary.**

I work in Artificial Intelligence, more specifically in Multi-agent Systems and Software Engineering, but I have ventured into the areas of Robotics, Ubiquitous Computing, and the Internet of Things. In the management area, I work in technology dissemination (and currently social media), technological innovation, entrepreneurship, and Information Technology Governance and process mapping.

- I'm currently working on [multi-agent-system](#) [embedded-system](#) [modeling-language](#) [metamodel](#)
- I'm collaborating on [JaCaMo](#) [ChonIDE](#)

**Research Groups and Projects**

- Member of Cognitive Hardware on Networks Research Group (CHON).
- Co-coordinator of the Brazilian Investment Olympics (OBINVEST).
- Founder of the Turing Project.

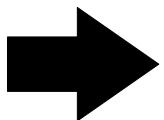
**Languages and frameworks I've interacted**

Java MySQL

**Follow me on Social Media**

in RG G ID Instagram

# GitHub



Screenshot of a GitHub profile page for [profpantoja](https://github.com/profpantoja).

The profile page shows the user's profile picture (a person with rainbow hair holding a paintbrush), name (**Kadu Pantoja**), and bio (**profpantoja**). It also lists **62 followers** and **8 following**.

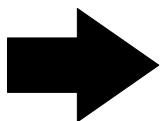
The GitHub interface includes a search bar, navigation tabs for Overview, Repositories, Projects, Packages, Stars, and a sidebar for Research Groups and Programming Languages.

Key repositories shown:

- java-exercicios** (Public): A repository for Java exercise lists, last updated yesterday.
- chonGame** (Private): A JavaFX game for learning object-oriented concepts, last updated 2 weeks ago.
- bd-exercicios** (Public): A repository for Database exercise lists, last updated on Dec 13, 2023.

The "Languages and frameworks I've interacted with" section lists Java, MySQL, and React.

The "Follow me on Social Media" section includes links to LinkedIn, GitHub, Google Scholar, and Instagram.



Screenshot of a GitHub profile page for [profpantoja](https://github.com/profpantoja).

The profile page shows the user's profile picture (a person with rainbow hair holding a paintbrush), name (**Kadu Pantoja**), bio, follower count (62), and links to CEFET/RJ, Rio de Janeiro, and a personal website.

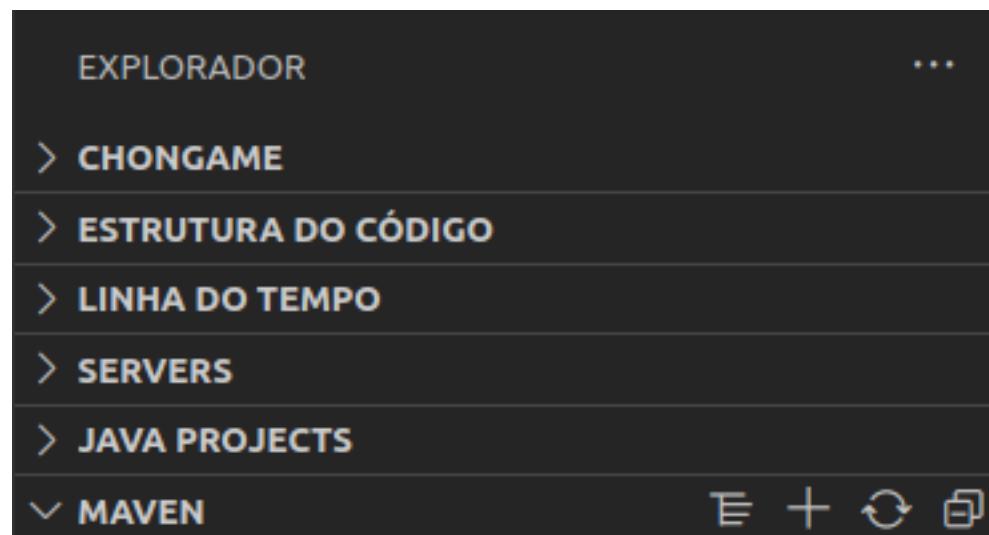
The main content area displays the user's repositories:

- java-exercicios** (Public)  
Descrição: Repertório das listas de Exercícios da disciplina de Linguagens e Técnicas de Programação II.  
Last updated: Updated yesterday.  
Languages: Java  
Contributors: 1
- chonGame** (Private)  
Descrição: A JavaFX game for learning the main concepts from the object-oriented approach.  
Last updated: Updated 2 weeks ago.  
Languages: Java
- bd-exercicios** (Public)  
Descrição: Repertório das listas de Exercícios da disciplina de Modelagem de Banco de Dados e Banco de Dados.  
Last updated: Updated on Dec 13, 2023.  
Languages: Java

The repository **chonGame** is highlighted with a red border.

At the bottom, there are sections for "Languages and frameworks I've interacted with" (Java, MySQL) and "Follow me on Social Media" (links to LinkedIn, ResearchGate, Google Scholar, ID, and Instagram).

# Creating a New Maven Project



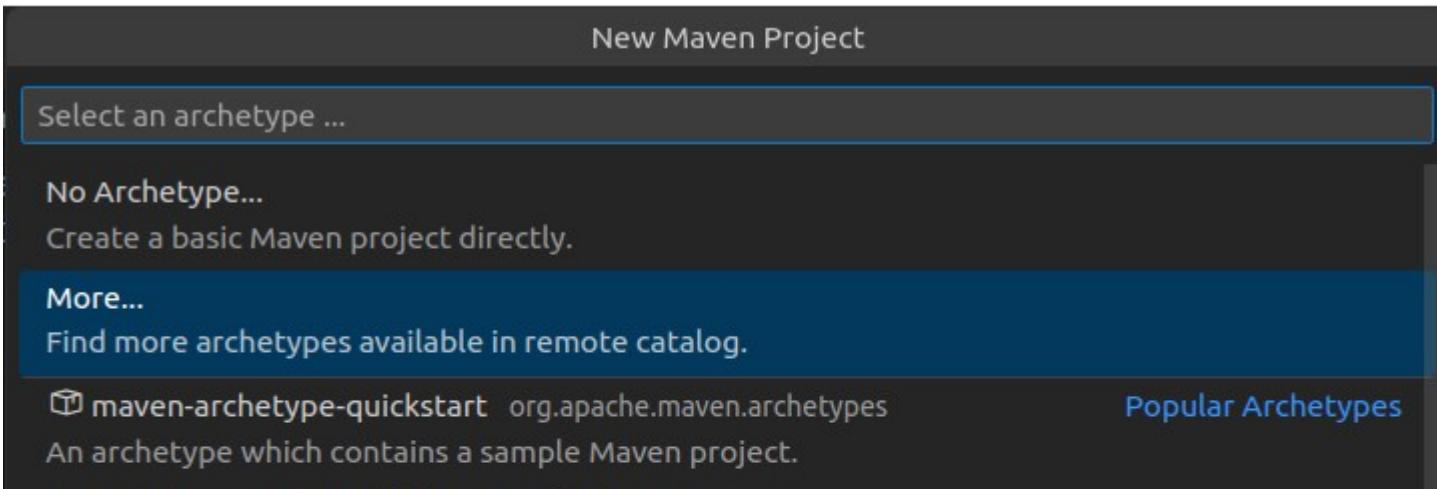
# Creating a New Maven Project



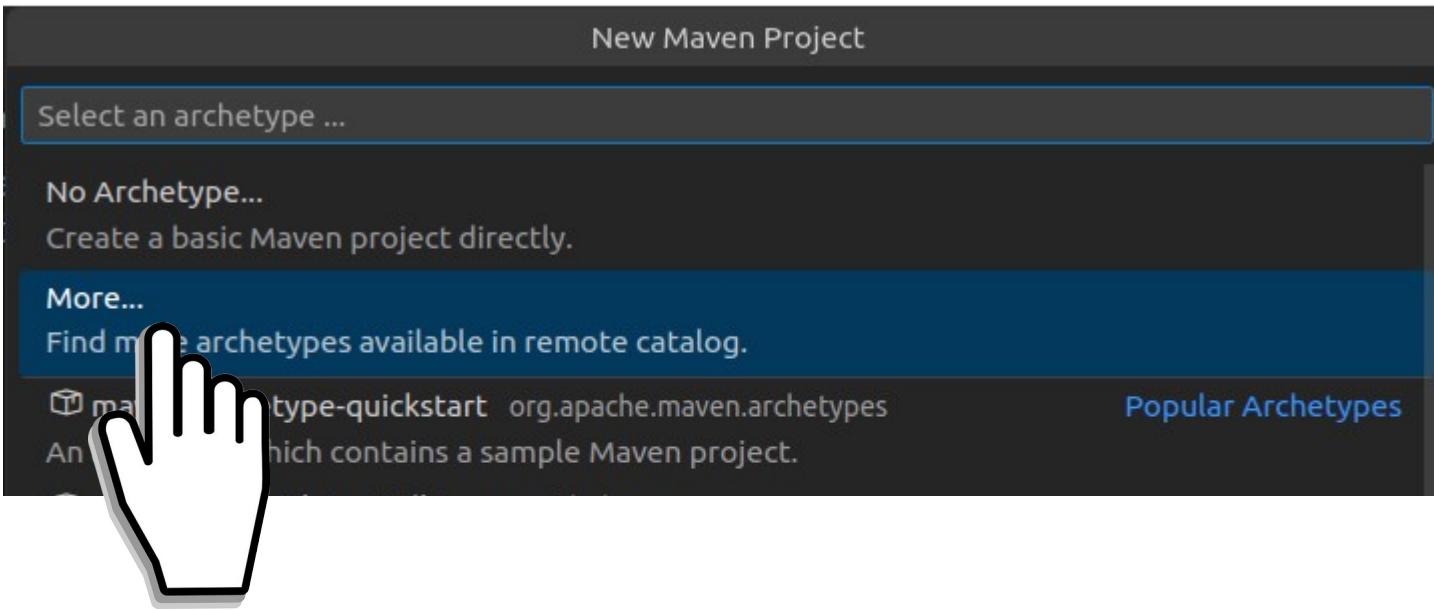
# Creating a New Maven Project



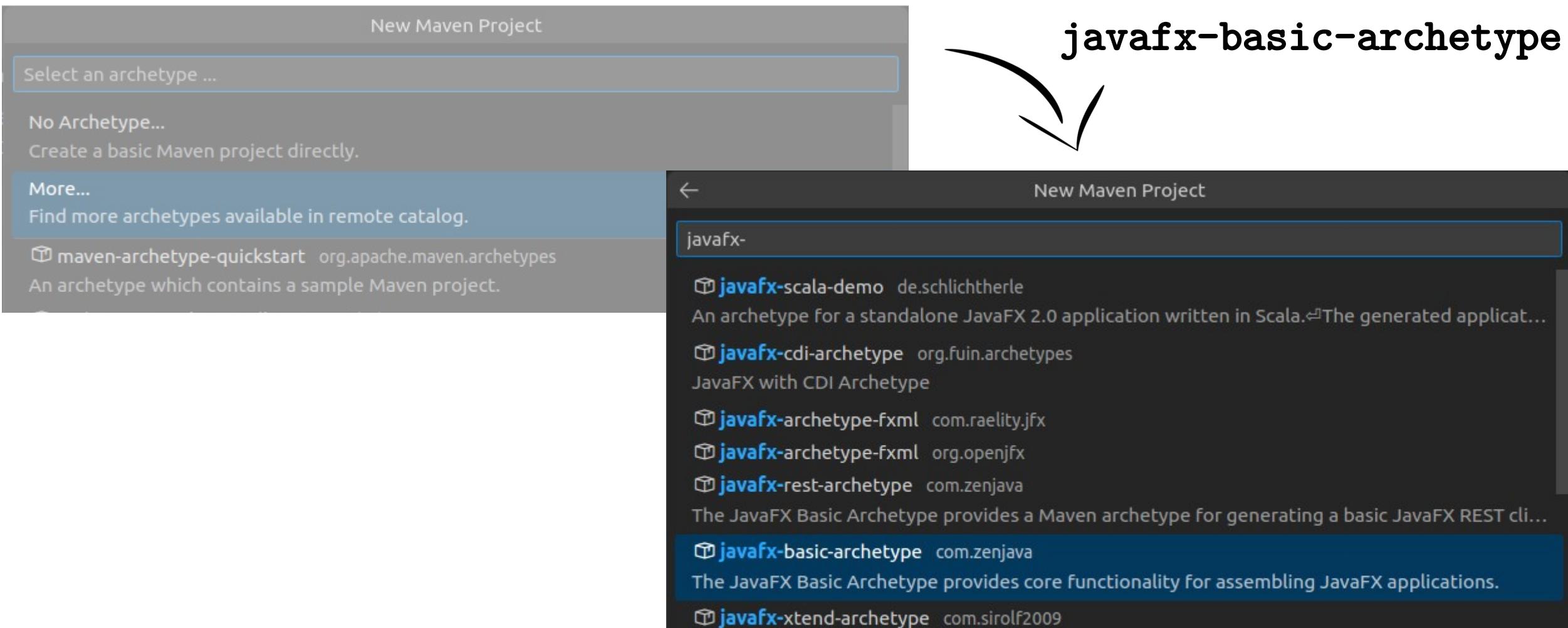
# Creating a New Maven Project



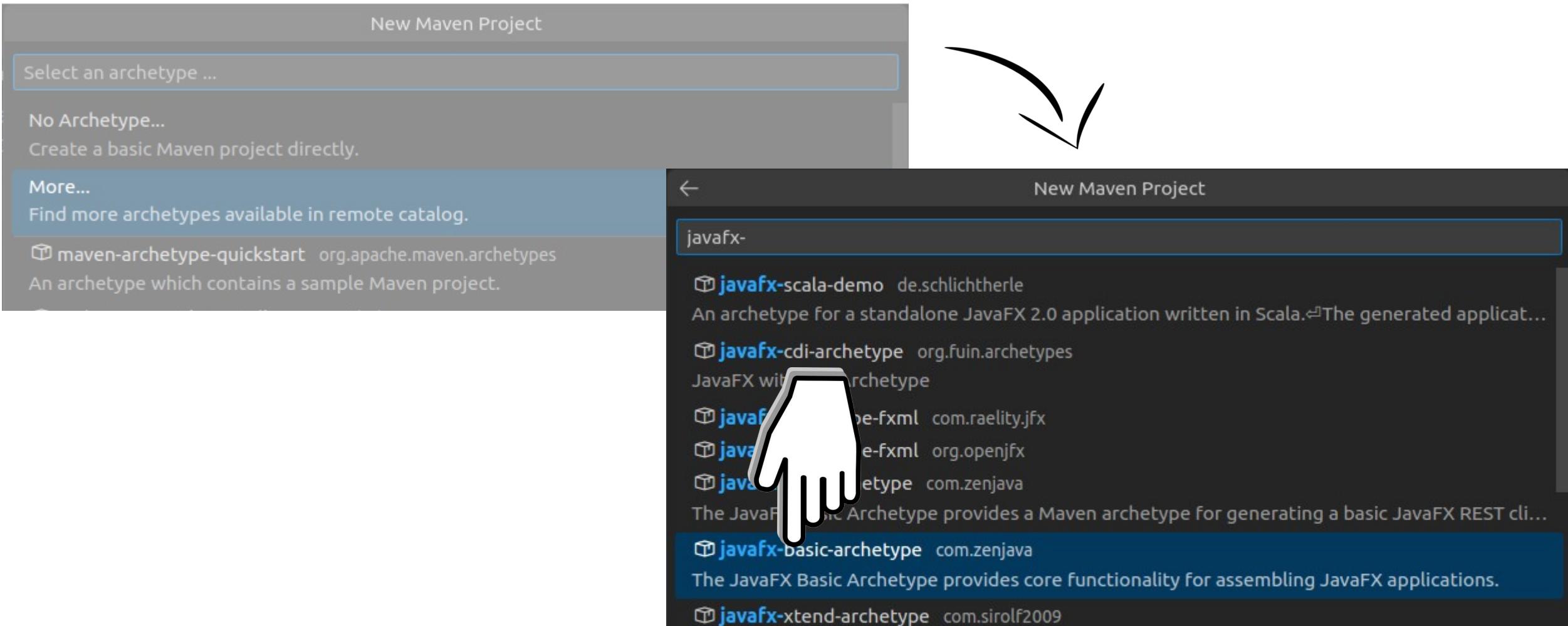
# Creating a New Maven Project



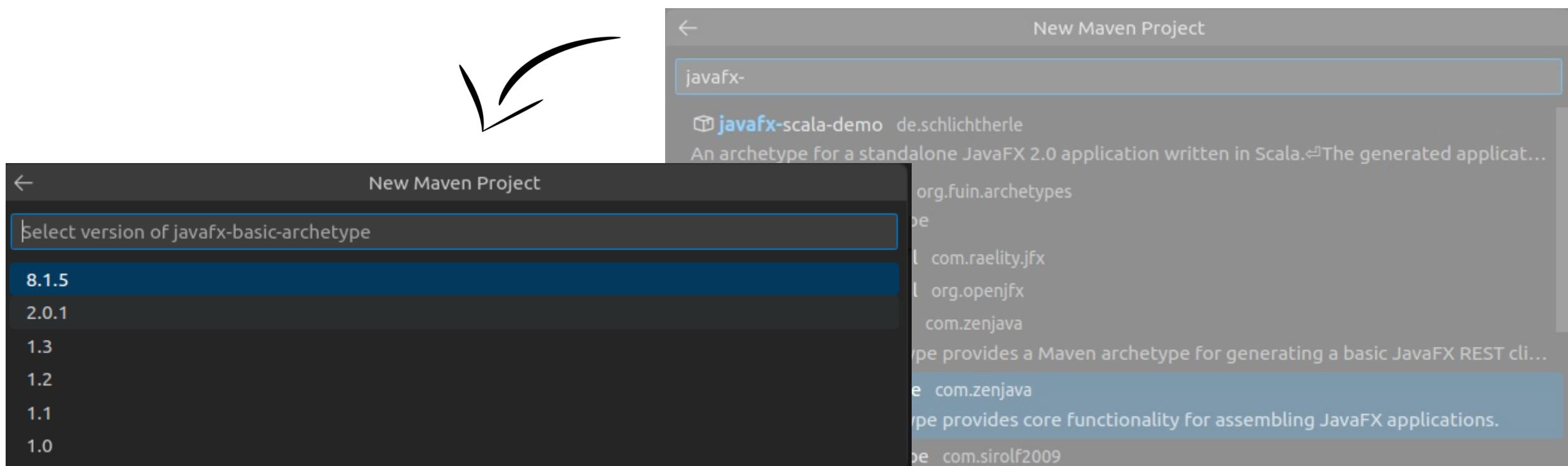
# Creating a New Maven Project



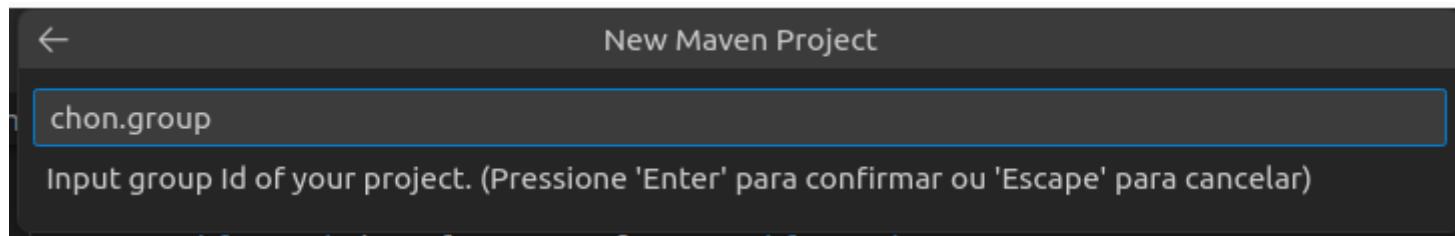
# Creating a New Maven Project



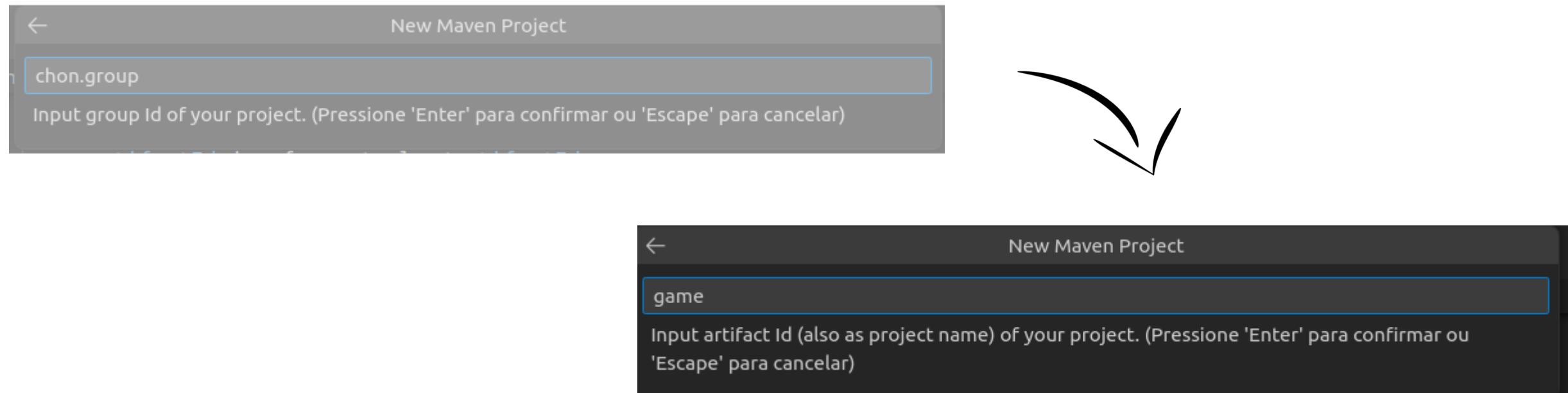
# Creating a New Maven Project



# Creating a New Maven Project



# Creating a New Maven Project



# Creating a New Maven Project

```
○ * Executando tarefa: mvn org.apache.maven.plugins:maven-archetype-plugin:3.1.2:generate -DarchetypeArtifactId="javafx-basi c-archetype" -DarchetypeGroupId="com.zenjava" -DarchetypeVersion="8.1.5" -DgroupId="chon.group" -DartifactId="game"

[INFO] Scanning for projects...
[INFO]
[INFO] -----< chon.group:game >-----
[INFO] Building game 1
[INFO] -----[ jar ]-----
[INFO]
[INFO] >>> maven-archetype-plugin:3.1.2:generate (default-cli) > generate-sources @ game >>
[INFO]
[INFO] <<< maven-archetype-plugin:3.1.2:generate (default-cli) < generate-sources @ game <<<
[INFO]
[INFO]
[INFO] --- maven-archetype-plugin:3.1.2:generate (default-cli) @ game ---
[INFO] Generating project in Interactive mode
[INFO] Archetype repository not defined. Using the one from [com.zenjava/javafx-basic-archetype:8.1.5] found in catalog remote
Downloading from central: https://repo.maven.apache.org/maven2/com/zenjava/javafx-basic-archetype/8.1.5/javafx-basic-archety pe-8.1.5.pom
Downloaded from central: https://repo.maven.apache.org/maven2/com/zenjava/javafx-basic-archetype/8.1.5/javafx-basic-archetyp e-8.1.5.pom (6.2 kB at 17 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/com/zenjava/javafx-basic-archetype/8.1.5/javafx-basic-archety pe-8.1.5.jar
Downloaded from central: https://repo.maven.apache.org/maven2/com/zenjava/javafx-basic-archetype/8.1.5/javafx-basic-archetyp e-8.1.5.jar (145 kB at 188 kB/s)
[INFO] Using property: groupId = chon.group
[INFO] Using property: artifactId = game
Define value for property 'version' 1.0-SNAPSHOT: :
```

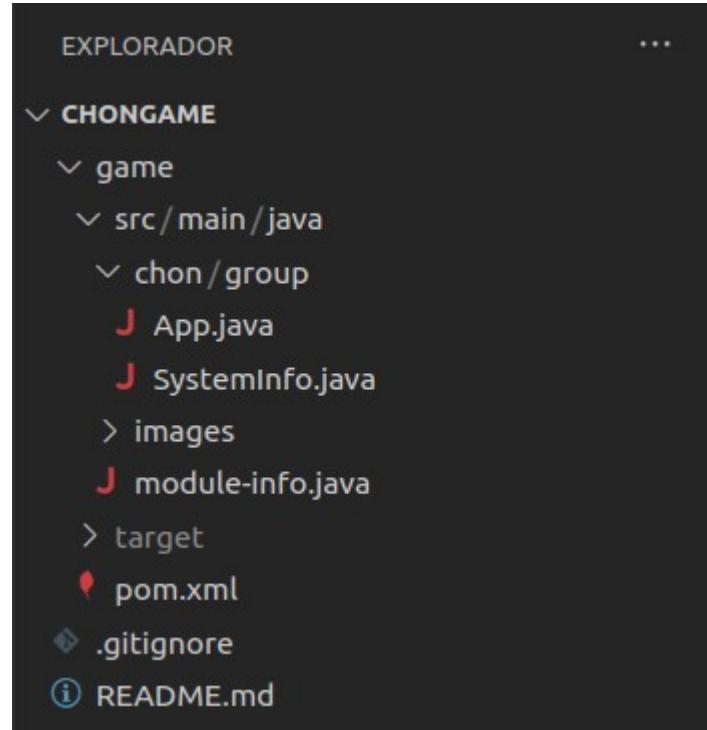
# Creating a New Maven Project

```
○ * Executando tarefa: mvn org.apache.maven.plugins:maven-archetype-plugin:3.1.2:generate -DarchetypeArtifactId="javafx-basi  
c-archetype" -DarchetypeGroupId="com.zenjava" -DarchetypeVersion="8.1.5" -DgroupId="chon.group" -DartifactId="game"  
  
[INFO] Scanning for projects...  
[INFO]  
[INFO] -----< chon.group:game >-----  
[INFO] Building game 1  
[INFO] -----[ jar ]-----  
[INFO]  
[INFO] >>> maven-archetype-plugin:3.1.2:generate (default-cli) > generate-sources @ game >>>  
[INFO]  
[INFO] <<< maven-archetype-plugin:3.1.2:generate (default-cli) < generate-sources @ game <<<  
[INFO]  
[INFO]  
[INFO] --- maven-archetype-plugin:3.1.2:generate (default-cli) @ game ---  
[INFO] Generating project in Interactive mode  
[INFO] Archetype repository not defined. Using the one from [com.zenjava/javafx-basic-archetype:8.1.5] found in catalog remote  
Downloading from central: https://repo.maven.apache.org/maven2/com/zenjava/javafx-basi  
pe-8.1.5.pom  
Downloaded from central: https://repo.maven.apache.org/maven2/com/zenjava/javafx-basi  
e-8.1.5.pom (6.2 kB at 17 kB/s)  
Downloading from central: https://repo.maven.apache.org/maven2/com/zenjava/javafx-basi  
pe-8.1.5.jar  
Downloaded from central: https://repo.maven.apache.org/maven2/com/zenjava/javafx-basi  
e-8.1.5.jar (145 kB at 188 kB/s)  
[INFO] Using property: groupId = chon.group  
[INFO] Using property: artifactId = game  
Define value for property 'version' 1.0-SNAPSHOT: :
```

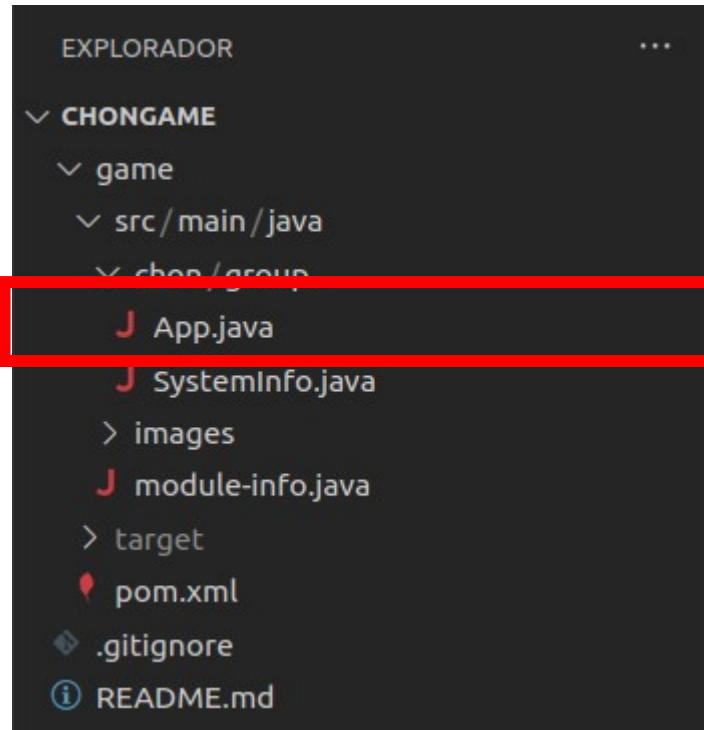


```
[INFO] Using property: groupId = chon.group  
[INFO] Using property: artifactId = game  
Define value for property 'version' 1.0-SNAPSHOT: : 1  
[INFO] Using property: package = chon.group  
Define value for property 'organizationName': chon  
Confirm properties configuration:  
groupId: chon.group  
artifactId: game  
version: 1  
package: chon.group  
organizationName: chon  
Y: : Y
```

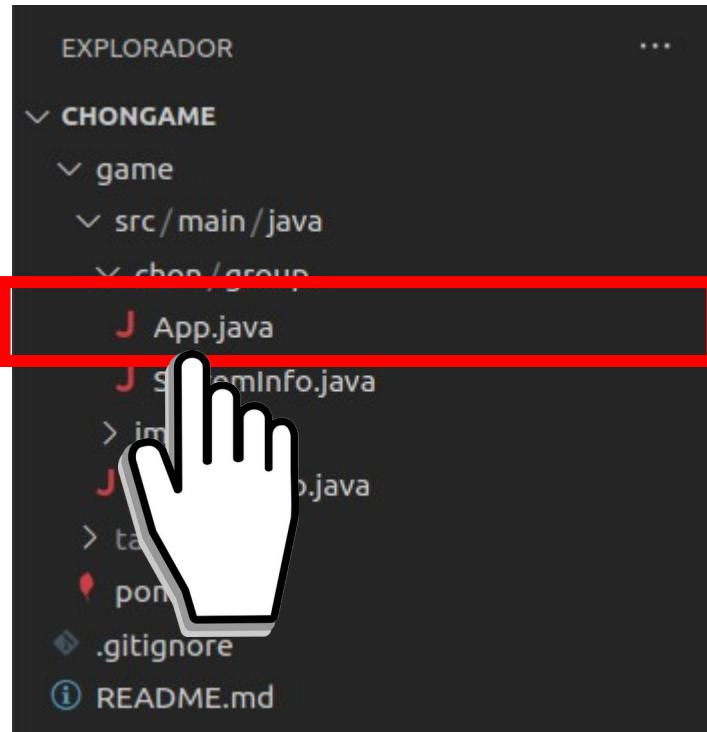
# Java FX App Running



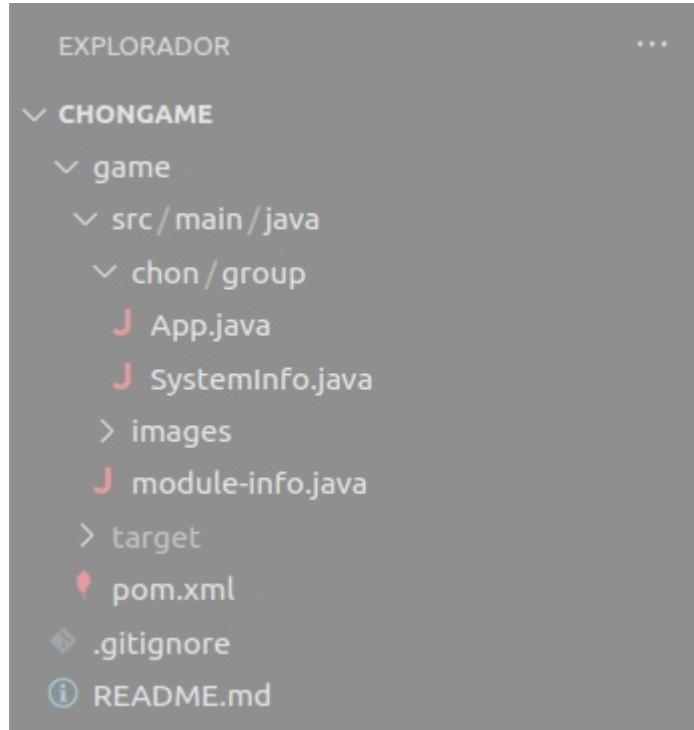
# Java FX App Running



# Java FX App Running



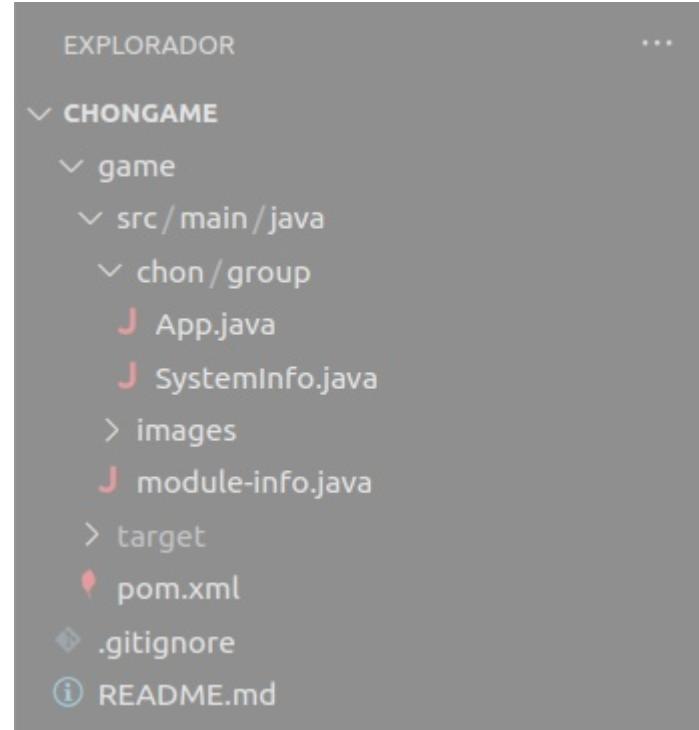
# Java FX App Running



The image shows a code editor window for 'App.java' with the following content:

```
App.java M X  
game > src > main > java > chon > group > J App.java > App > start(Stage)  
1 package chon.group;  
2  
3 import javafx.application.Application;  
4 import javafx.scene.Scene;  
5 import javafx.scene.control.Label;  
6 import javafx.scene.layout.StackPane;  
7 import javafx.stage.Stage;  
8  
9 /**  
10  * JavaFX App  
11 */  
12 public class App extends Application {  
13  
14     @Override  
15     public void start(Stage stage) {  
16         var javaVersion = SystemInfo.javaVersion();  
17         var javafxVersion = SystemInfo.javafxVersion();  
18  
19         var label = new Label("Hello, JavaFX " + javafxVersion + ", running on Java " + javaVersion + ".");  
20         var scene = new Scene(new StackPane(label), width:640, height:480);  
21         stage.setScene(scene);  
22         stage.show();  
23     }  
24  
25     public static void main(String[] args) {  
26         launch();  
27     }  
28 }  
29 }
```

# Java FX App Running

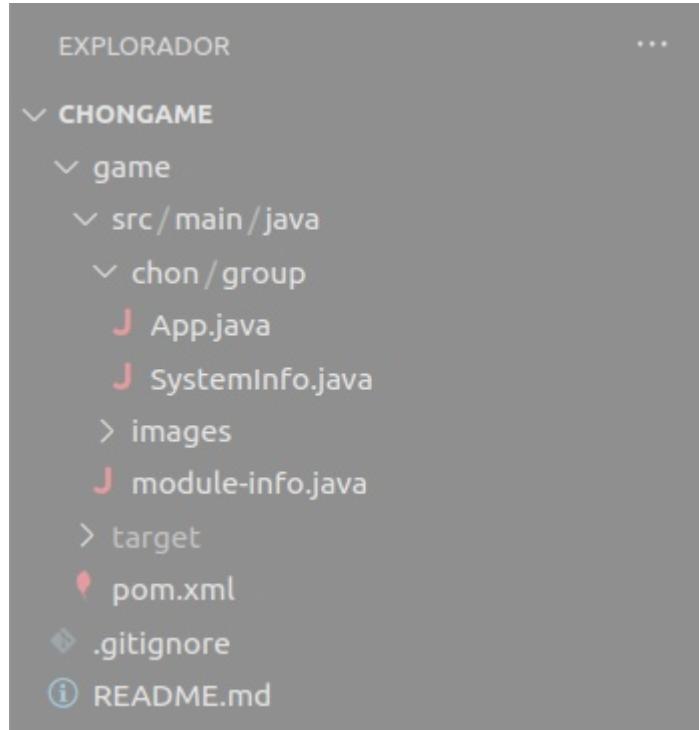


The image shows a code editor window with the file 'App.java' open. The code defines a JavaFX application that prints the Java and JavaFX versions to a window.

```
App.java M X
game > src > main > java > chon > group > J App.java > App > start(Stage)
1 package chon.group;
2
3 import javafx.application.Application;
4 import javafx.scene.Scene;
5 import javafx.scene.control.Label;
6 import javafx.scene.layout.StackPane;
7 import javafx.stage.Stage;
8
9 /**
10 * JavaFX App
11 */
12 public class App extends Application {
13
14     @Override
15     public void start(Stage stage) {
16         var javaVersion = SystemInfo.javaVersion();
17         var javafxVersion = SystemInfo.javafxVersion();
18
19         var label = new Label("Hello, JavaFX " + javafxVersion + ", running on Java " + javaVersion + ".");
20         var scene = new Scene(new StackPane(label), width:640, height:480);
21         stage.setScene(scene);
22         stage.show();
23     }
24
25     public static void main(String[] args) {
26         launch();
27     }
28 }
29 }
```

A red box highlights the 'Run | Debug' button located at the bottom of the code editor window.

# Java FX App Running

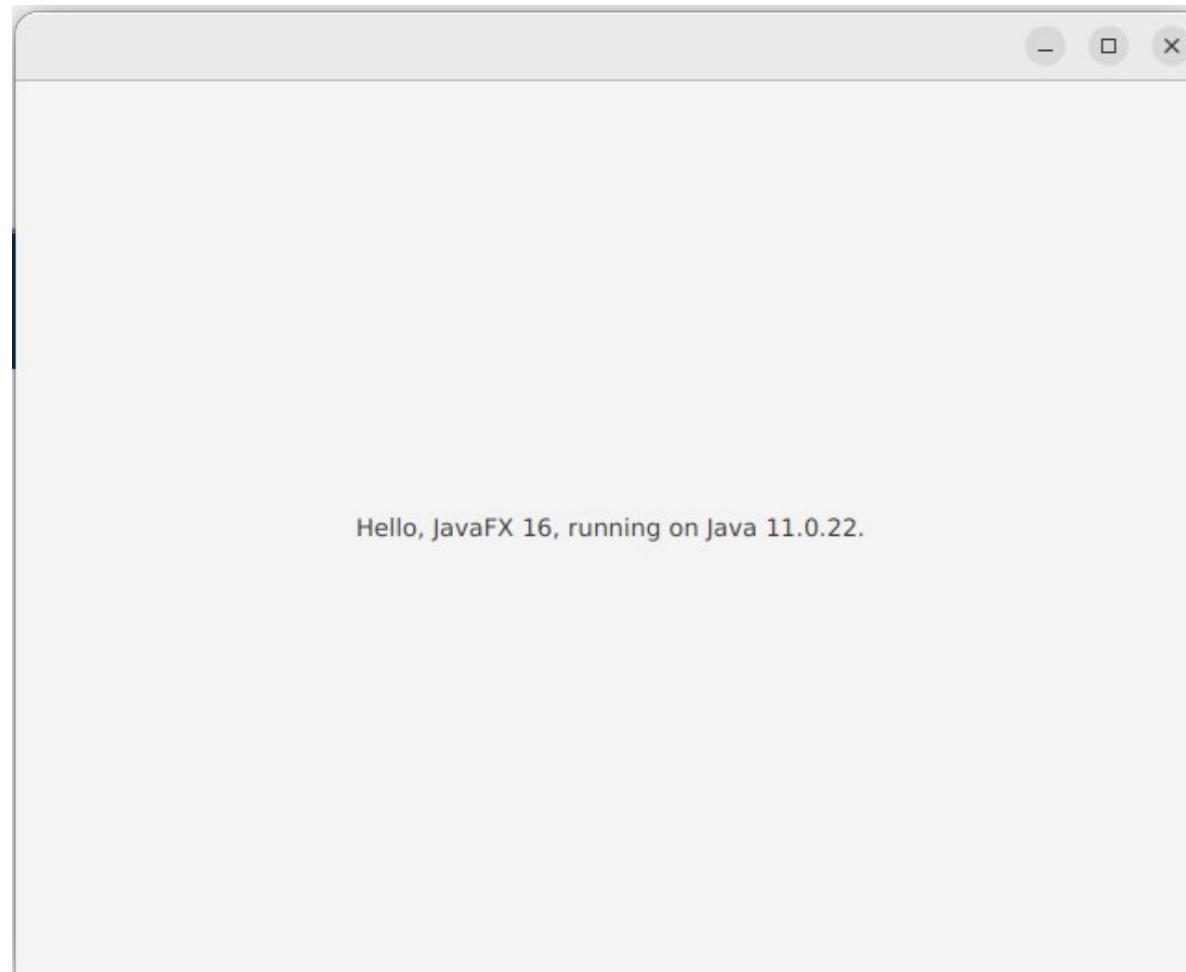


The code in 'App.java' is as follows:

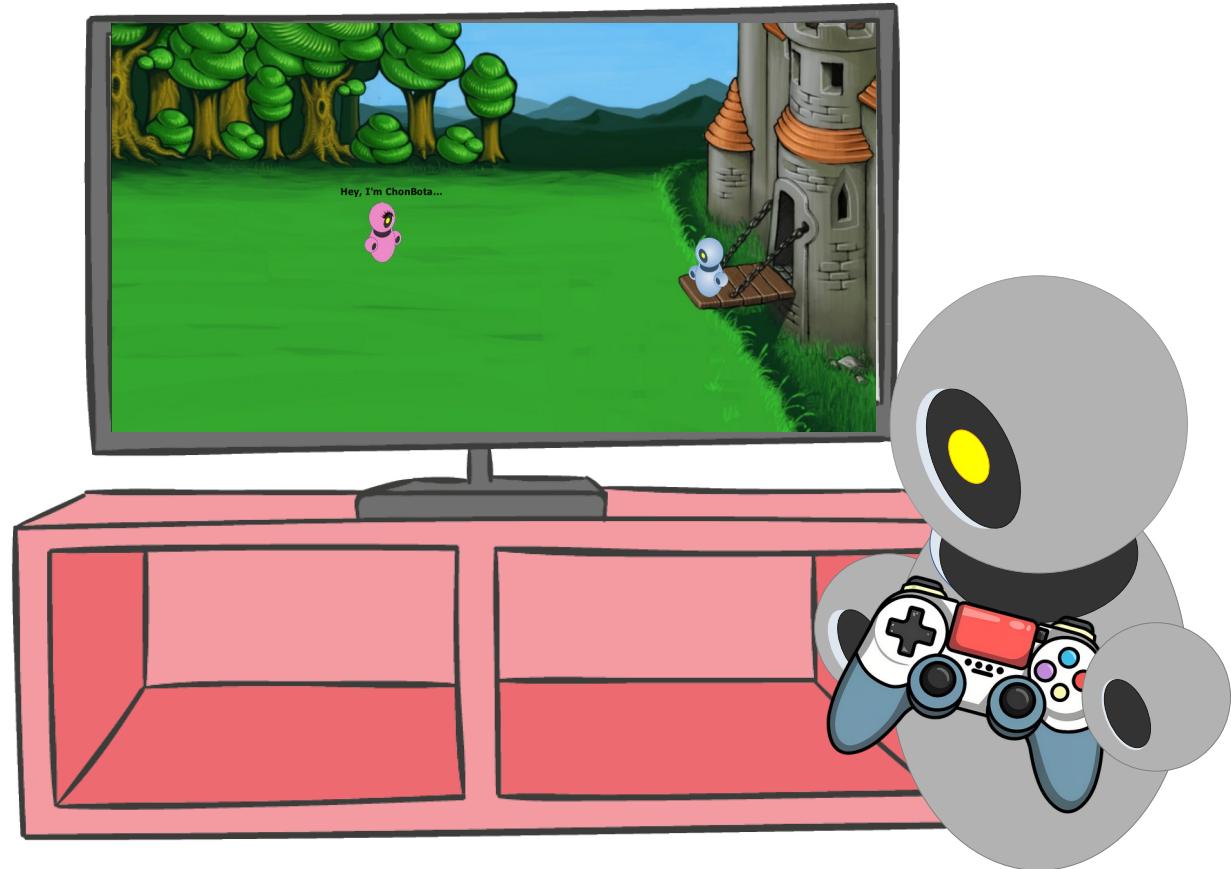
```
App.java M X
game > src > main > java > chon > group > J App.java > App > start(Stage)
1 package chon.group;
2
3 import javafx.application.Application;
4 import javafx.scene.Scene;
5 import javafx.scene.control.Label;
6 import javafx.scene.layout.StackPane;
7 import javafx.stage.Stage;
8
9 /**
10 * JavaFX App
11 */
12 public class App extends Application {
13
14     @Override
15     public void start(Stage stage) {
16         String javaVersion = SystemInfo.javaVersion();
17         String javafxVersion = SystemInfo.javafxVersion();
18
19         Label label = new Label("Hello, JavaFX " + javafxVersion + ", running on Java " + javaVersion + ".");
20         StackPane scene = new StackPane(label), width:640, height:480);
21         stage.setScene(scene);
22         stage.show();
23
24     }
25     public static void main(String[] args) {
26         launch();
27     }
28 }
29 }
```

A hand cursor icon is positioned over the code area, and a red rectangular box highlights the 'Run | Debug' button at the bottom of the code editor window.

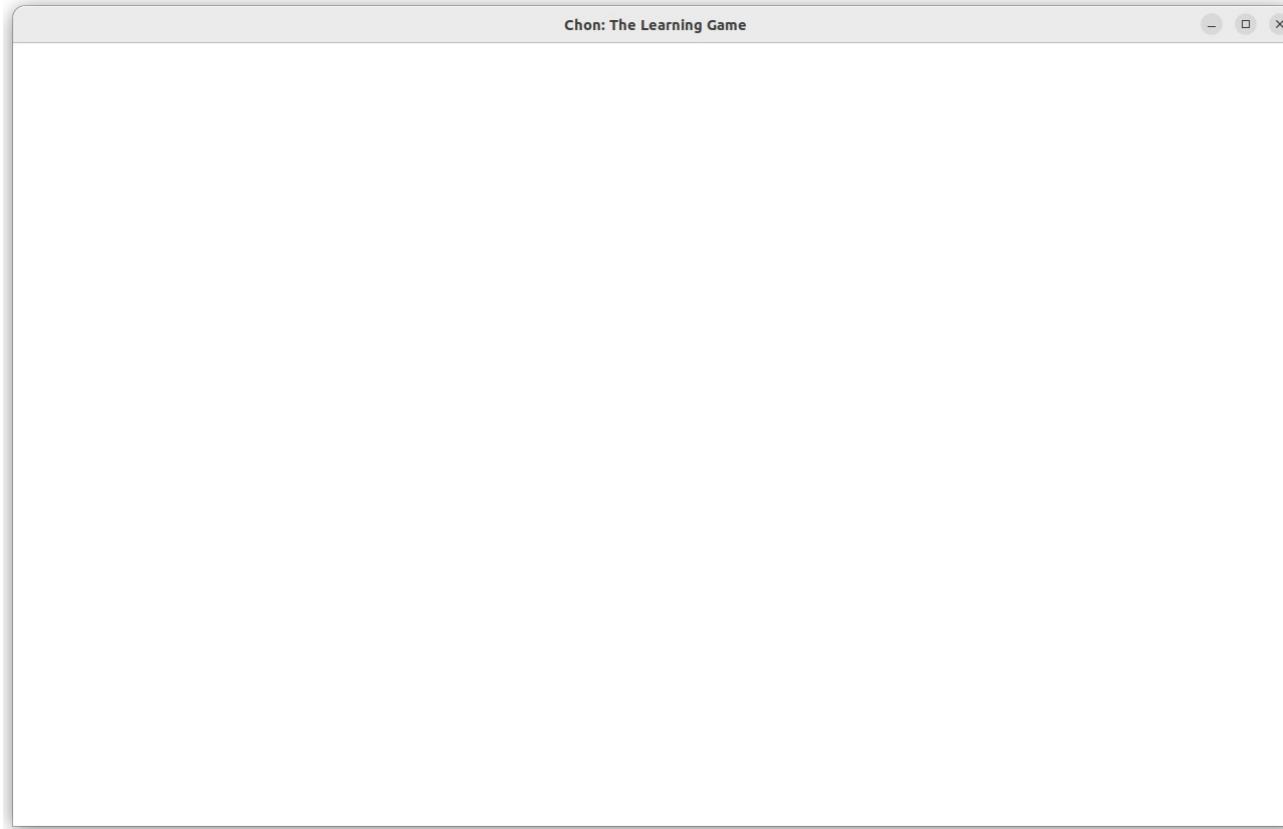
# Java FX App Running



# MANIPULATING GRAPHICAL ELEMENTS

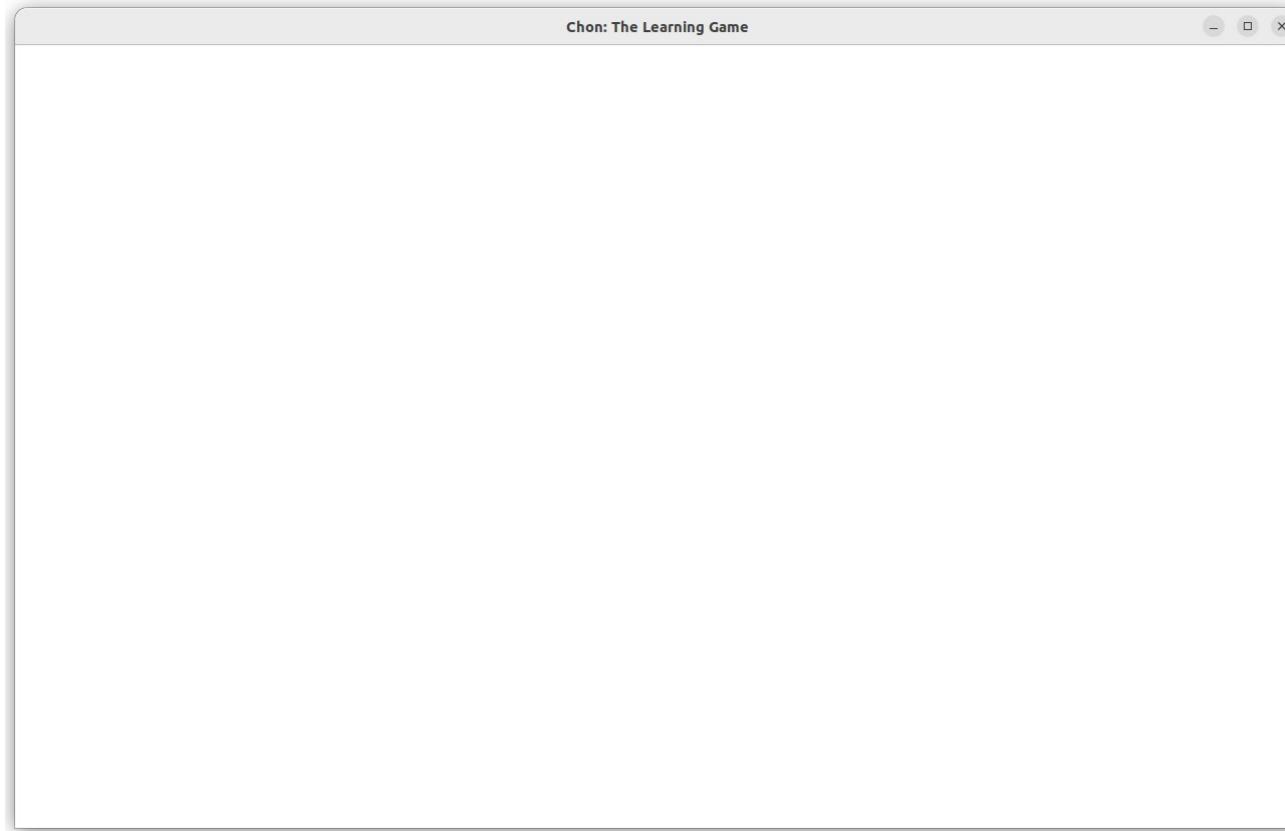


# StackPane



# StackPane

pane

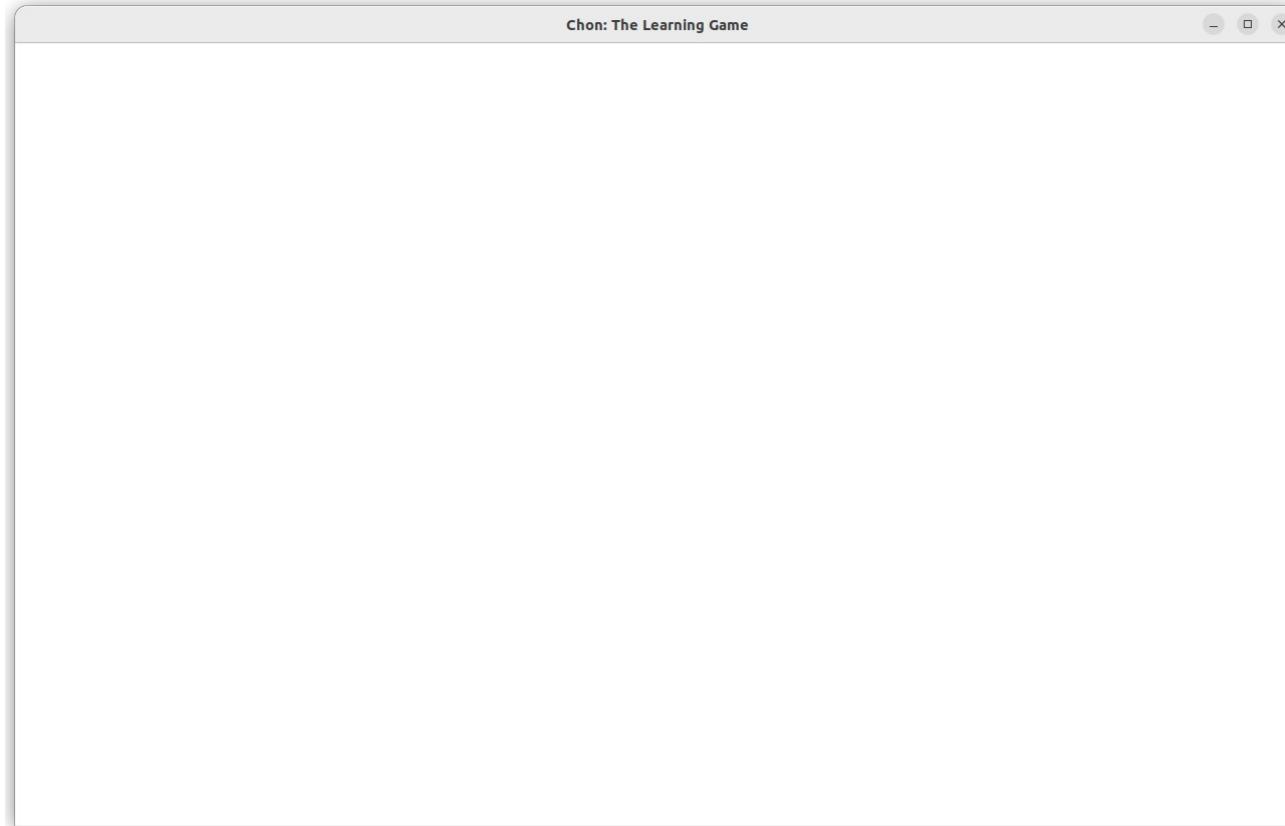


# StackPane

pane

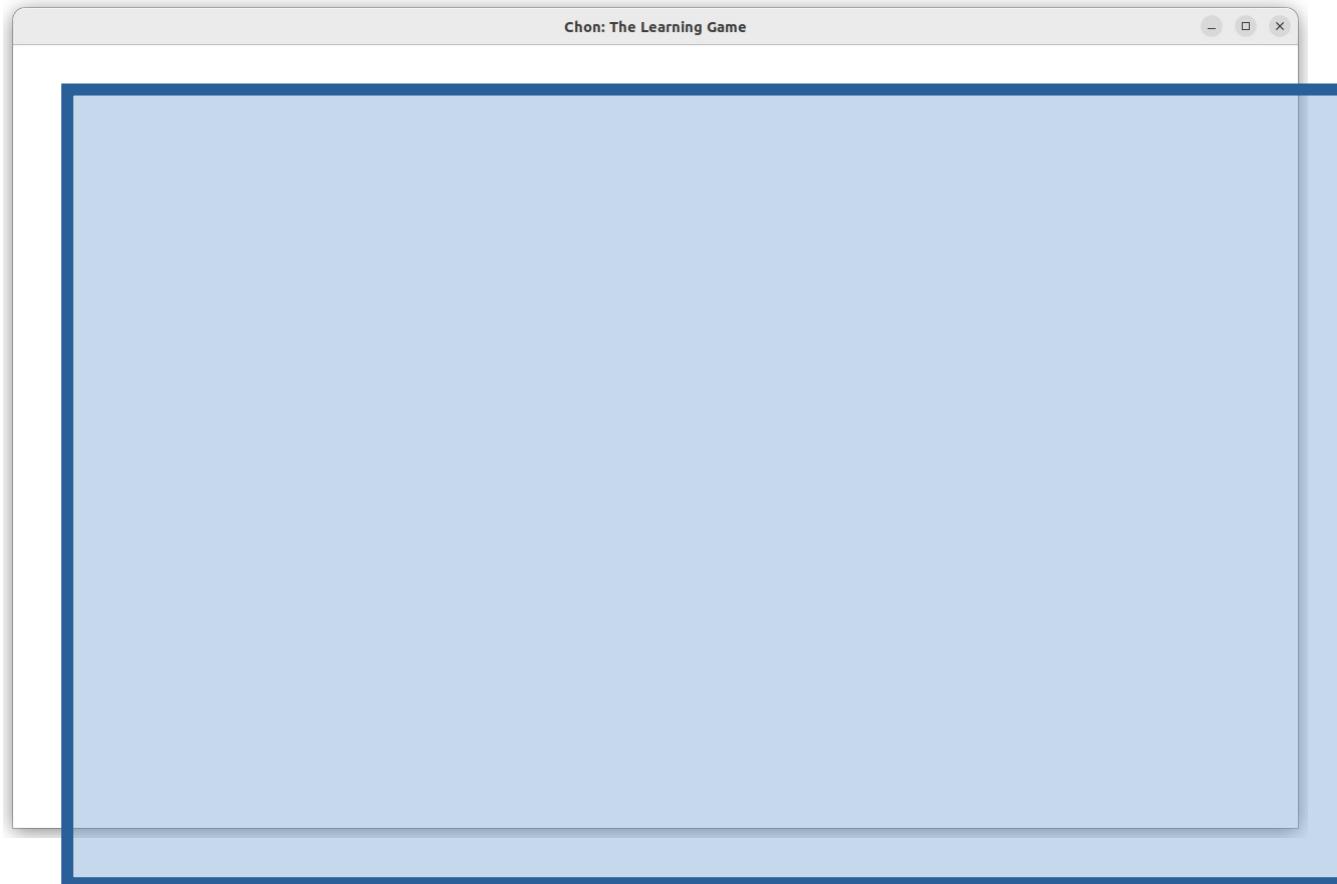


The window  
that is  
currently  
displayed on  
the screen.



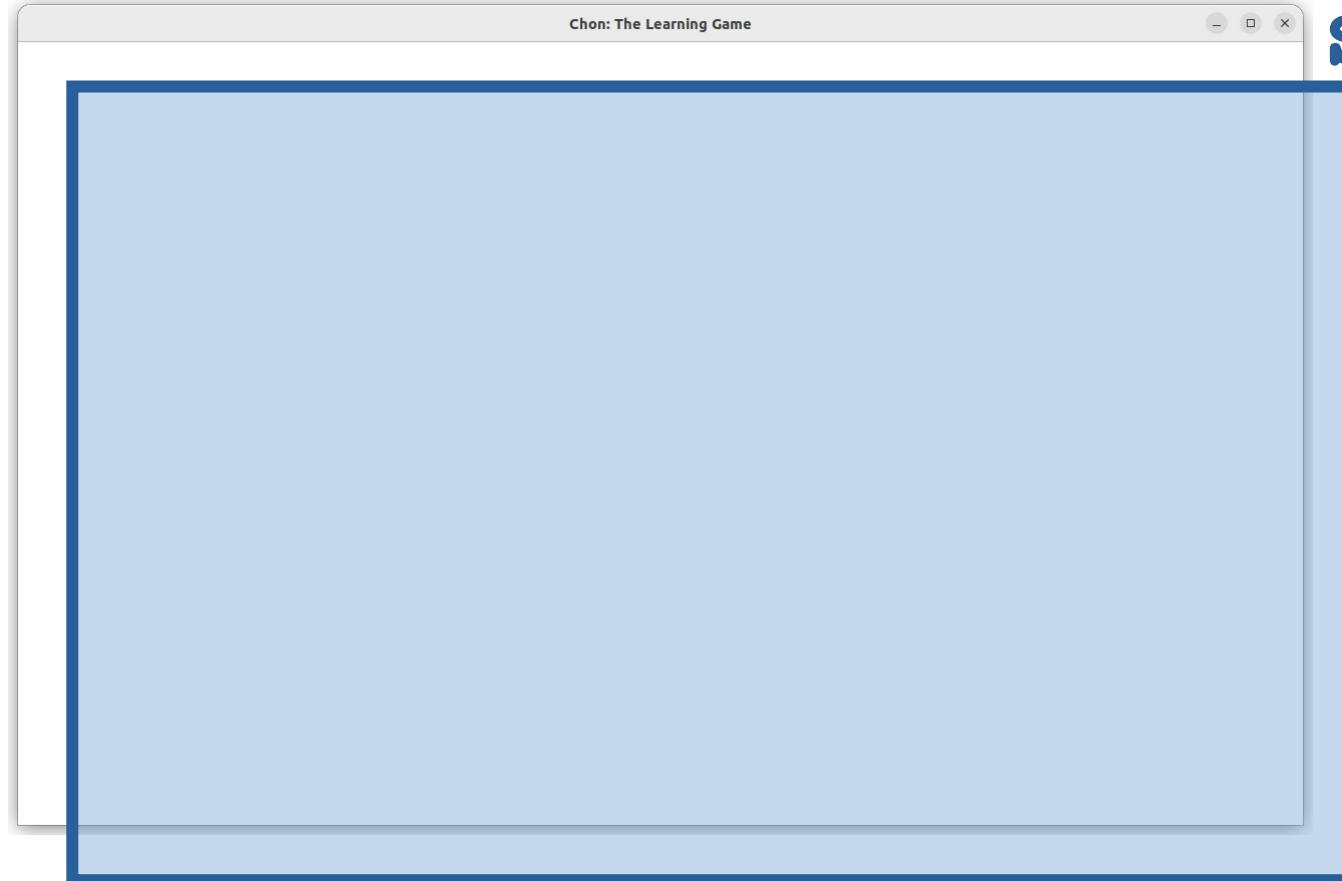
# Scene

pane



# Scene

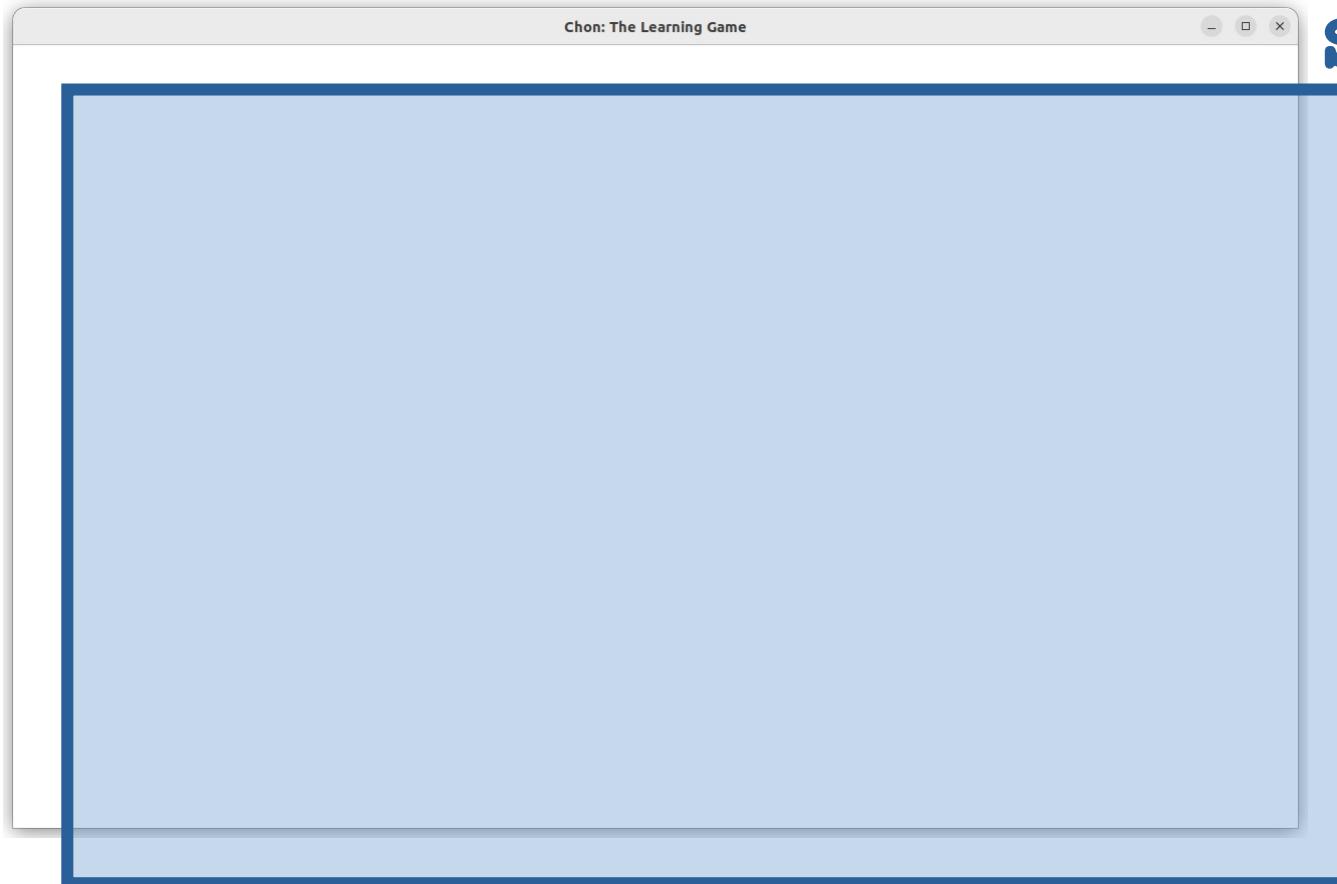
pane



scene

# Scene

pane



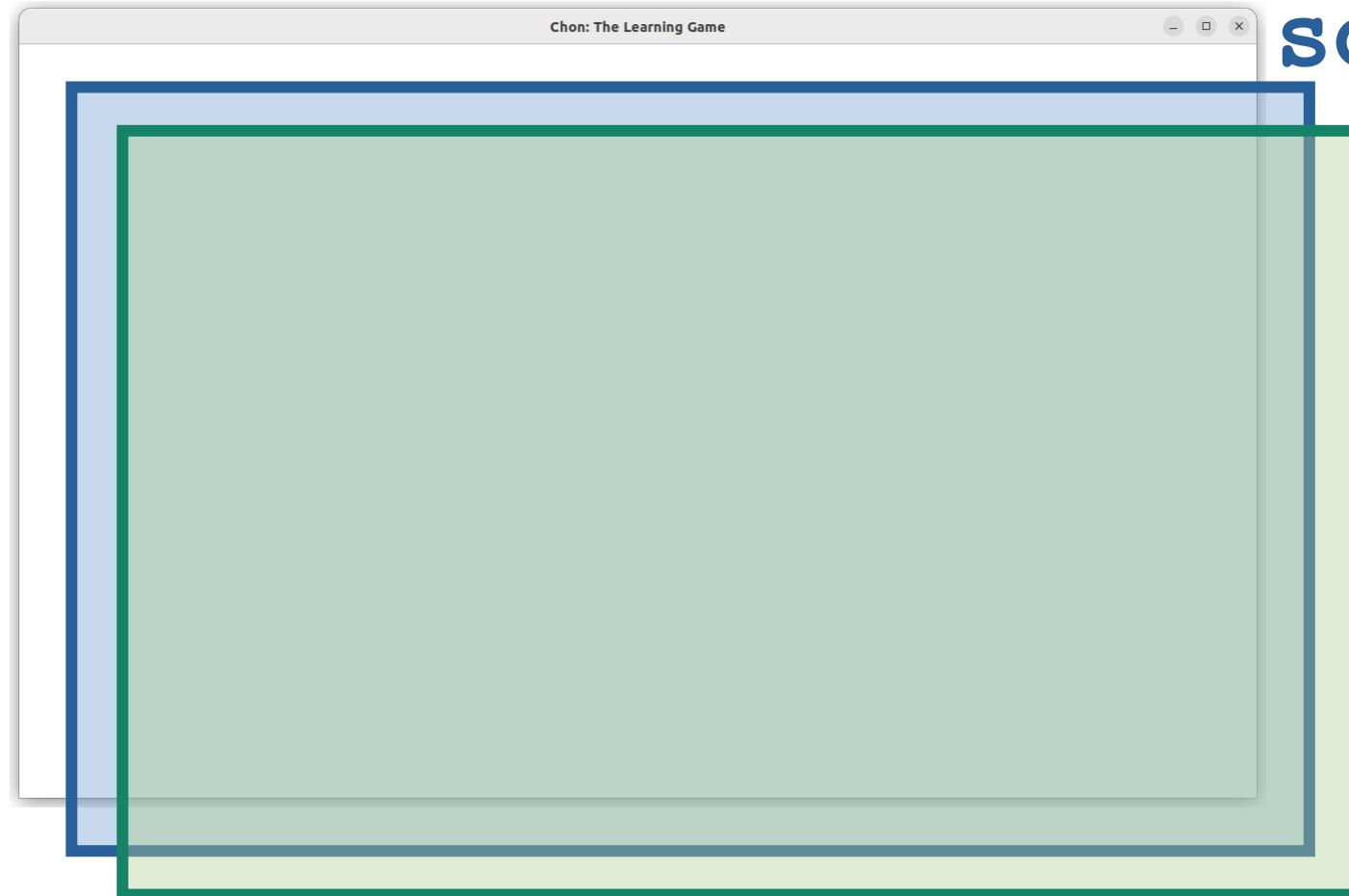
scene



It refers to  
the initial  
visible  
dimensions  
of a pane.

# Canvas

pane



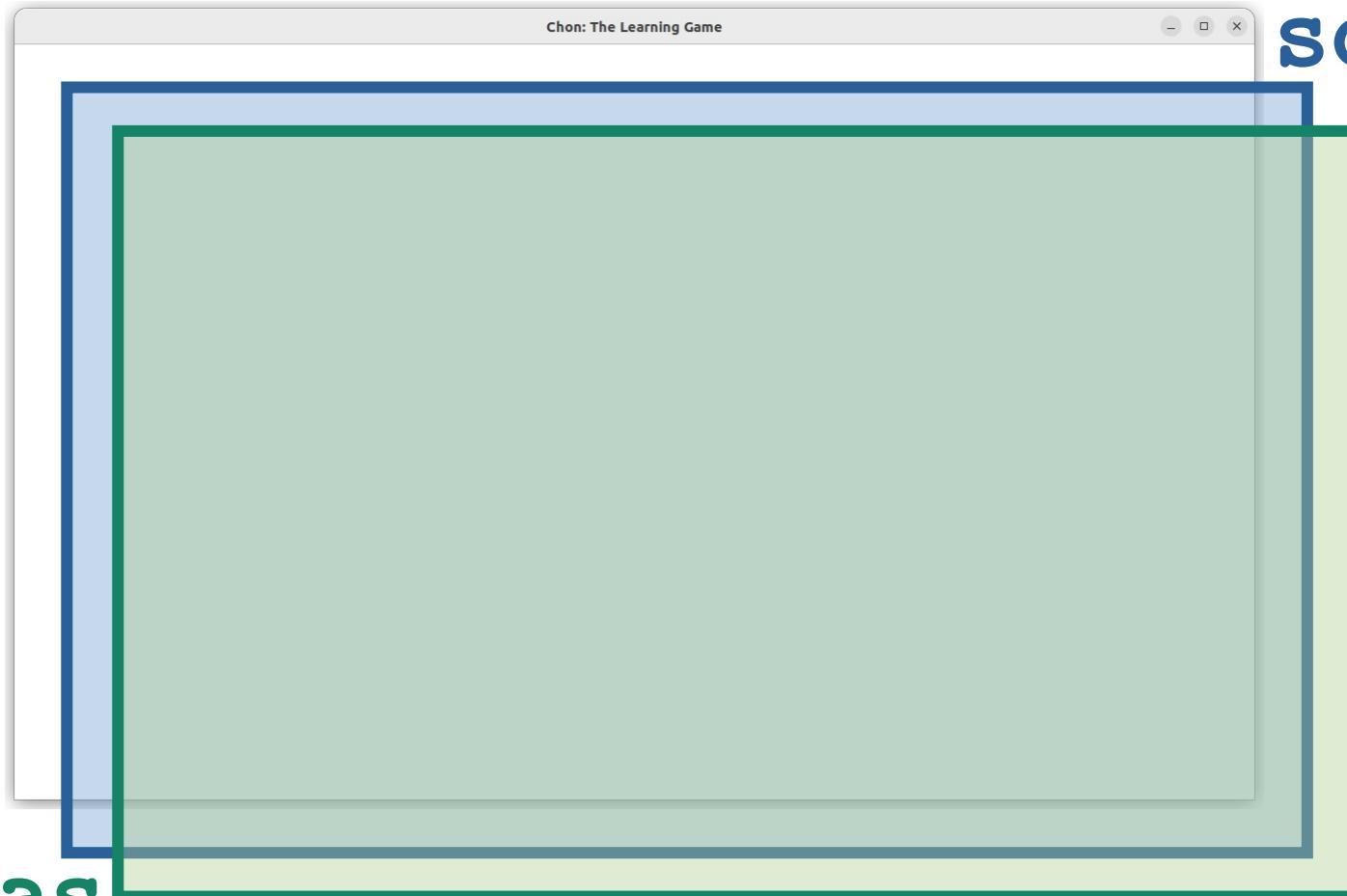
scene

# Canvas

pane

scene

canvas



# Canvas

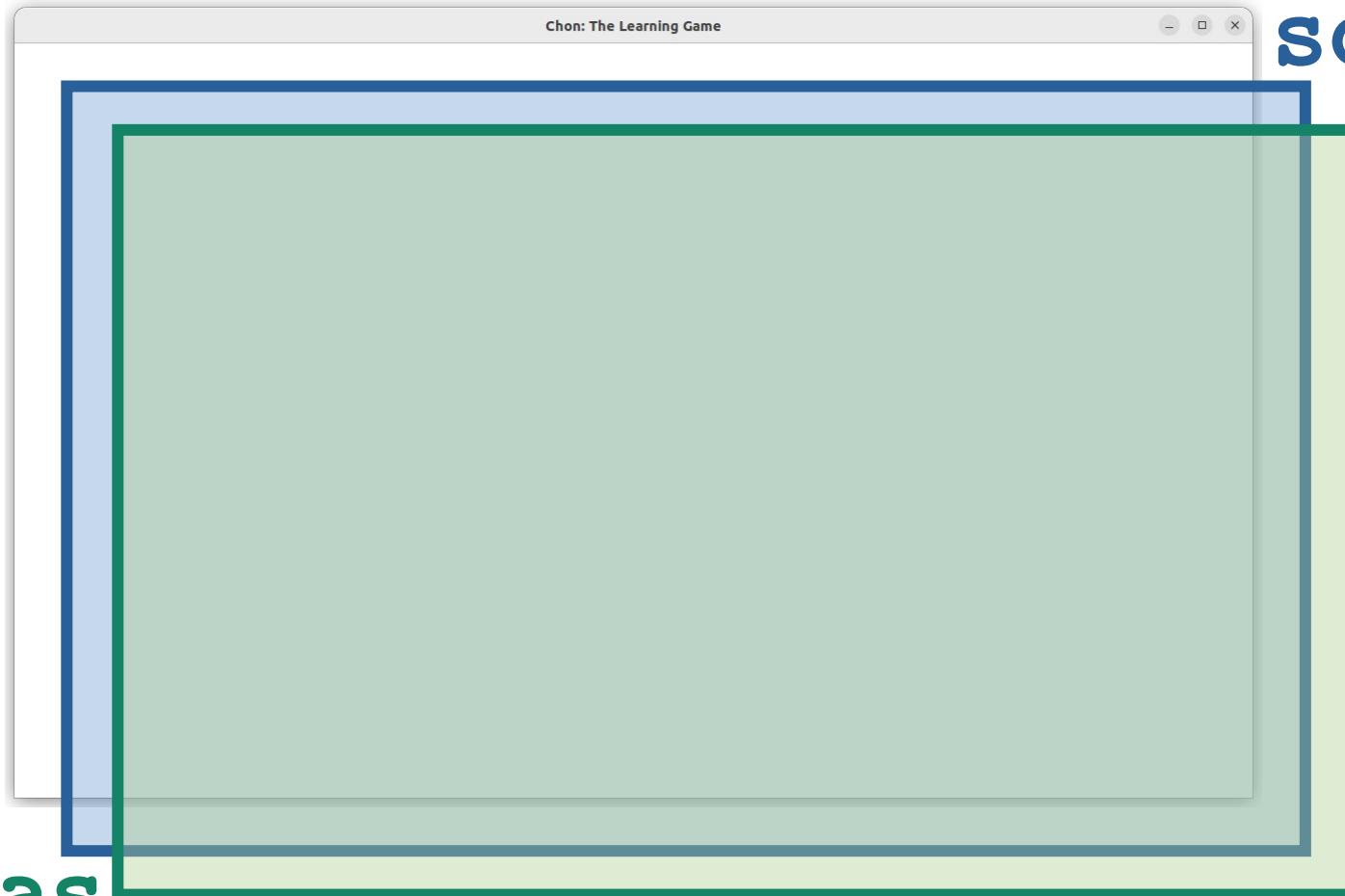
pane

The canvas is  
the part of  
the screen  
where graphics  
are displayed  
and printed.

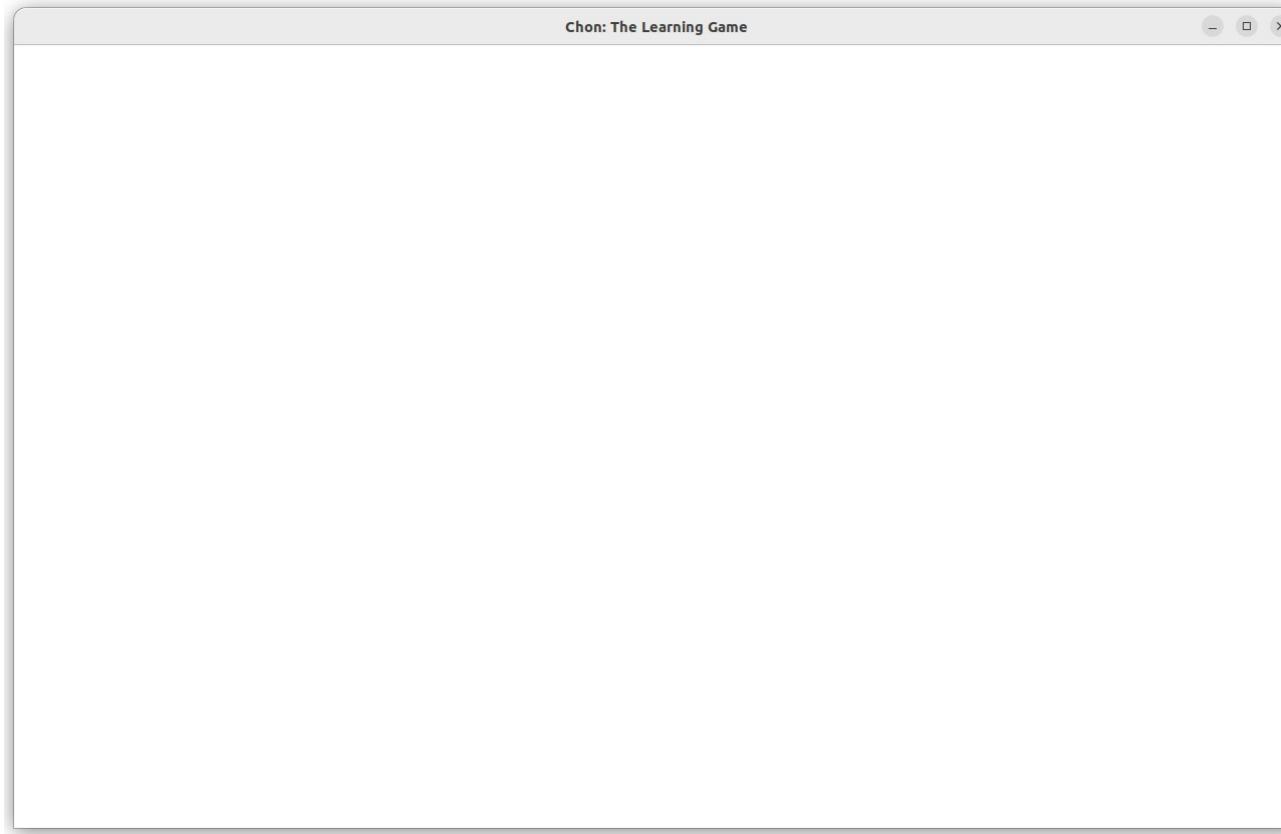


canvas

scene



# Object's Dimension: Width and Height

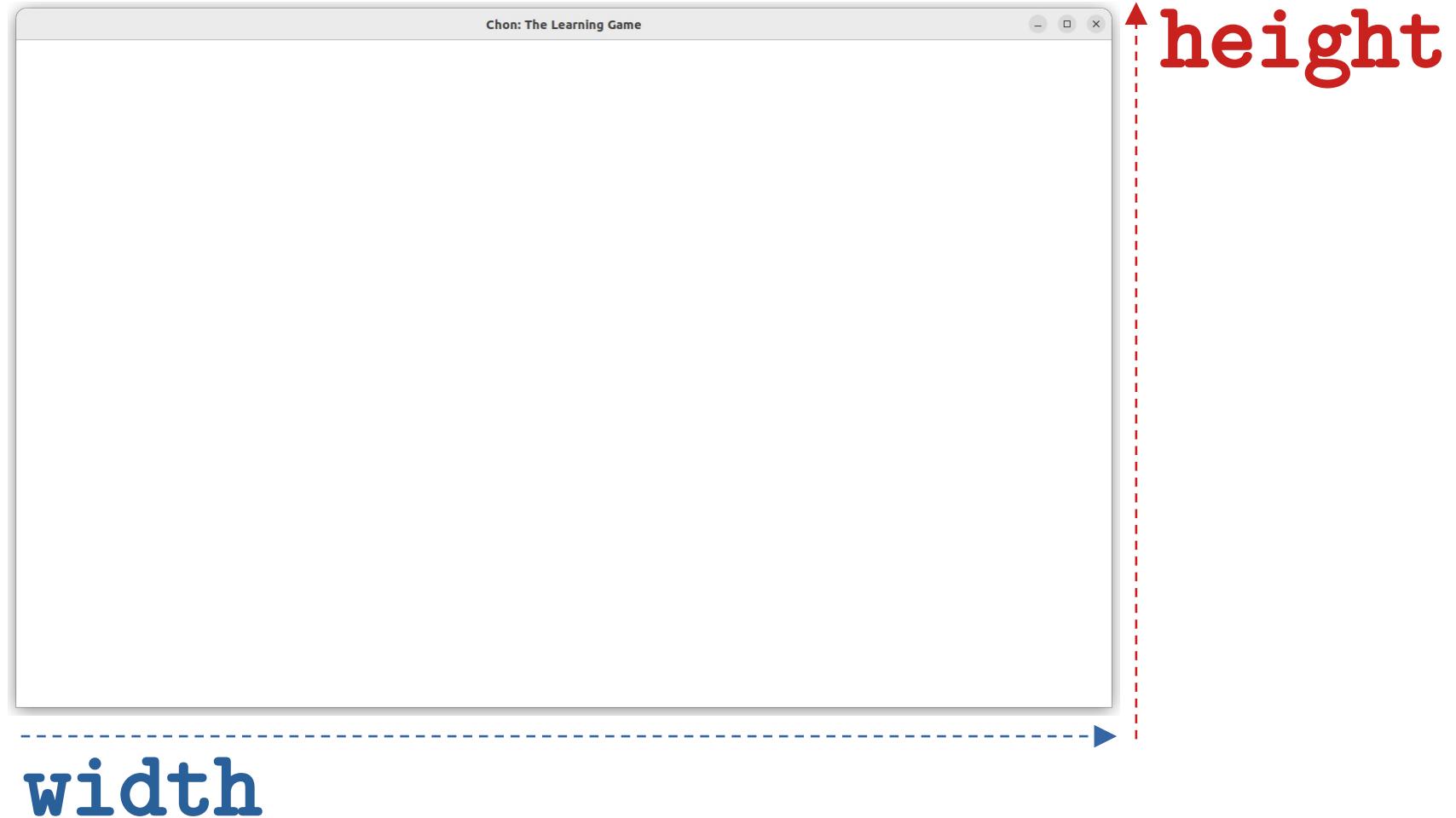


# Object's Dimension: Width and Height

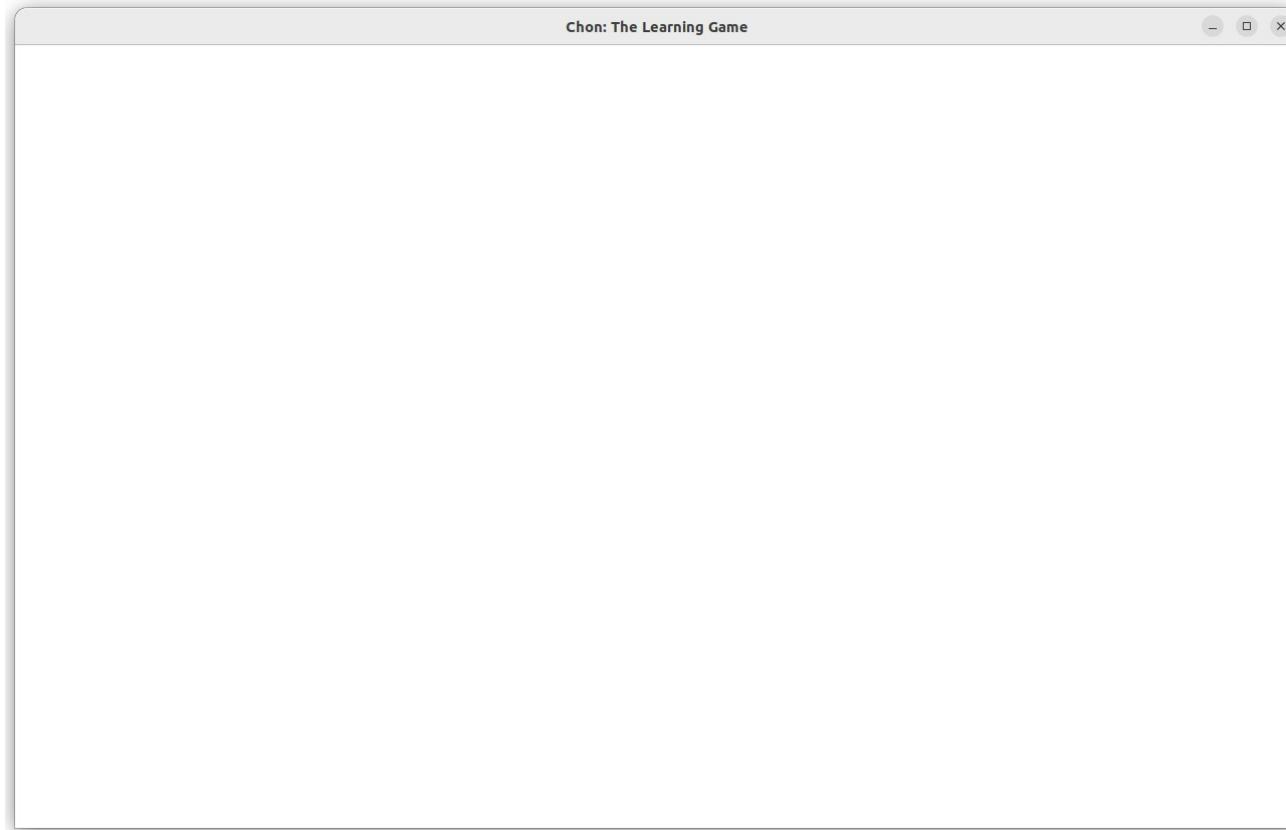


width

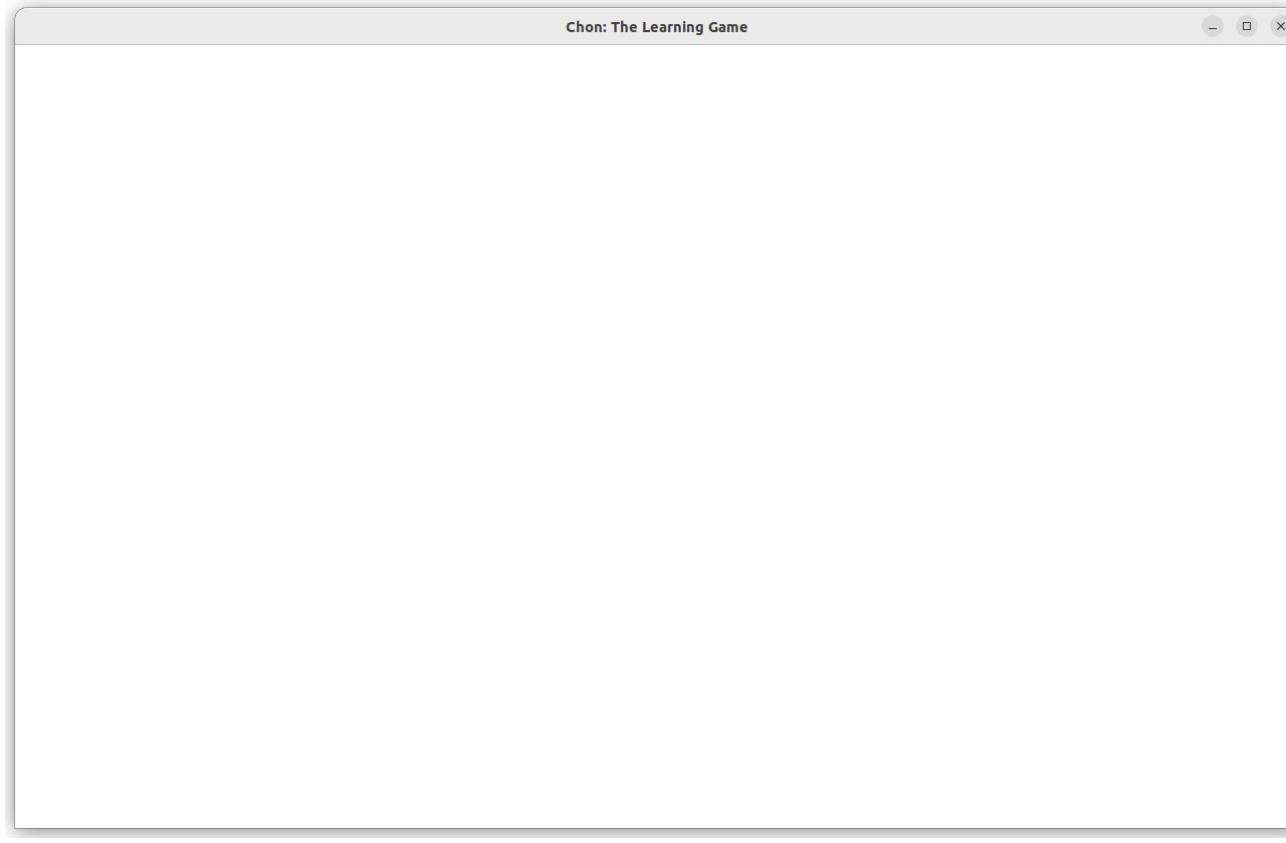
# Object's Dimension: Width and Height



# Object's Dimension: Width and Height

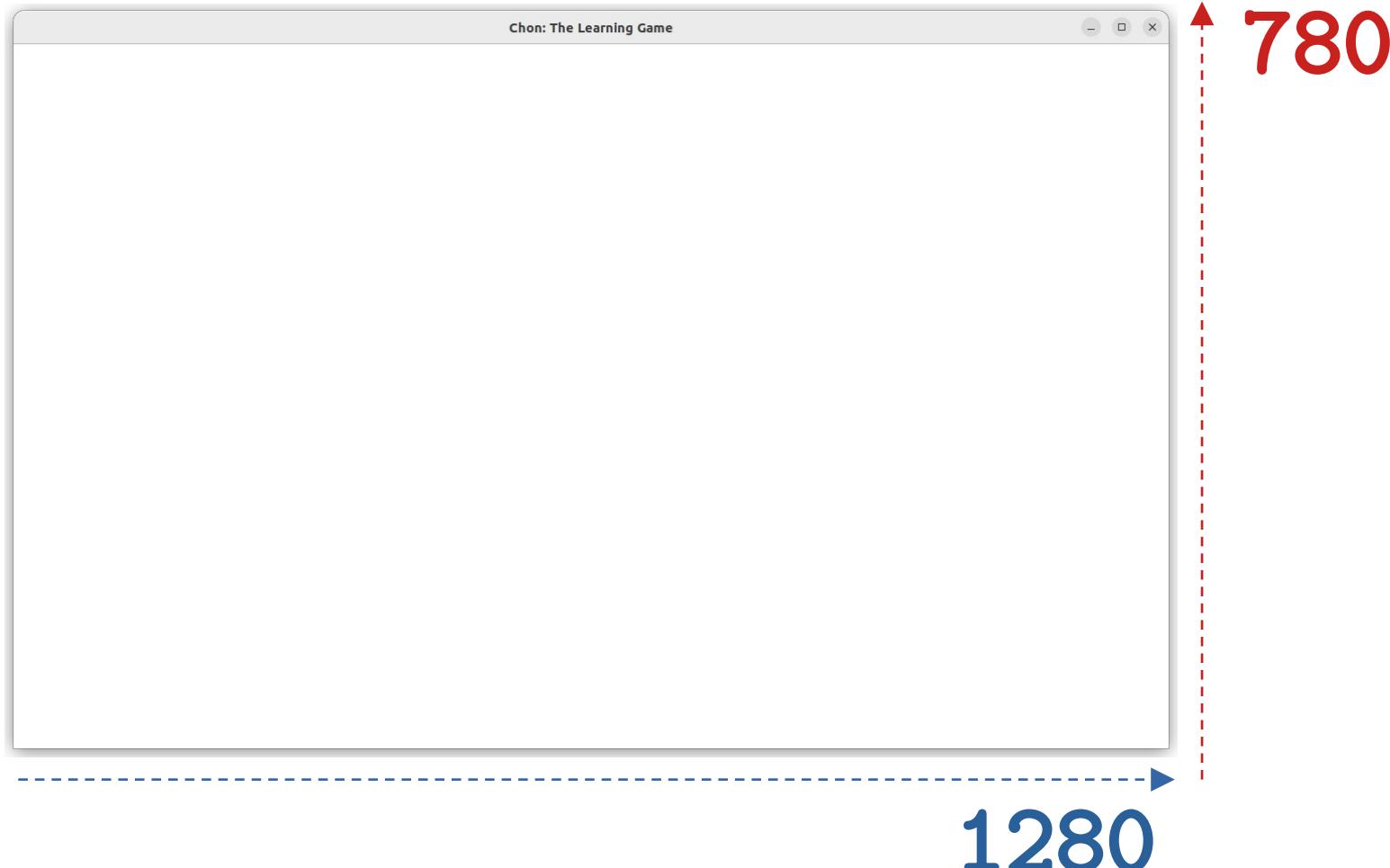


# Object's Dimension: Width and Height

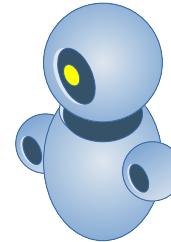
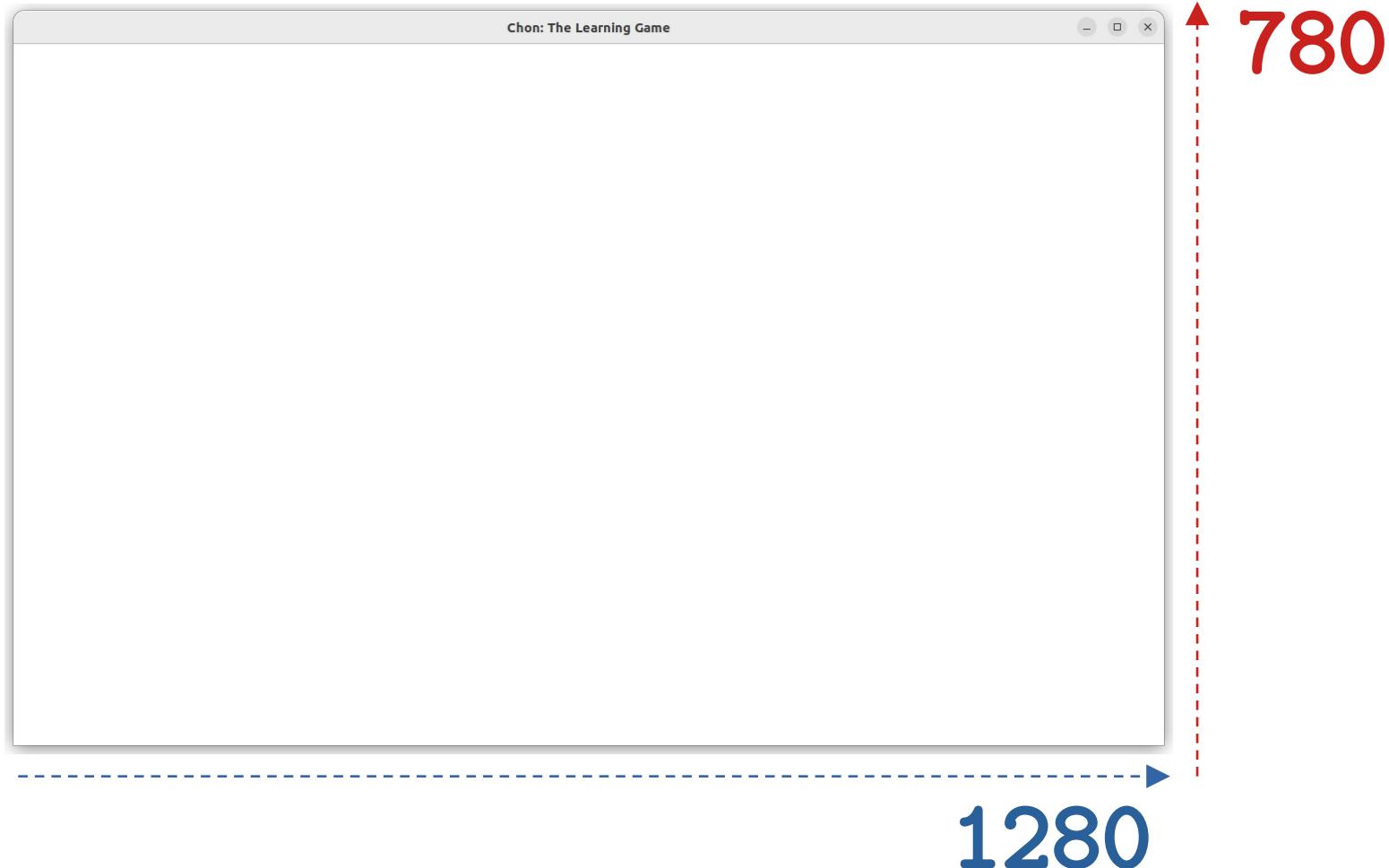


1280

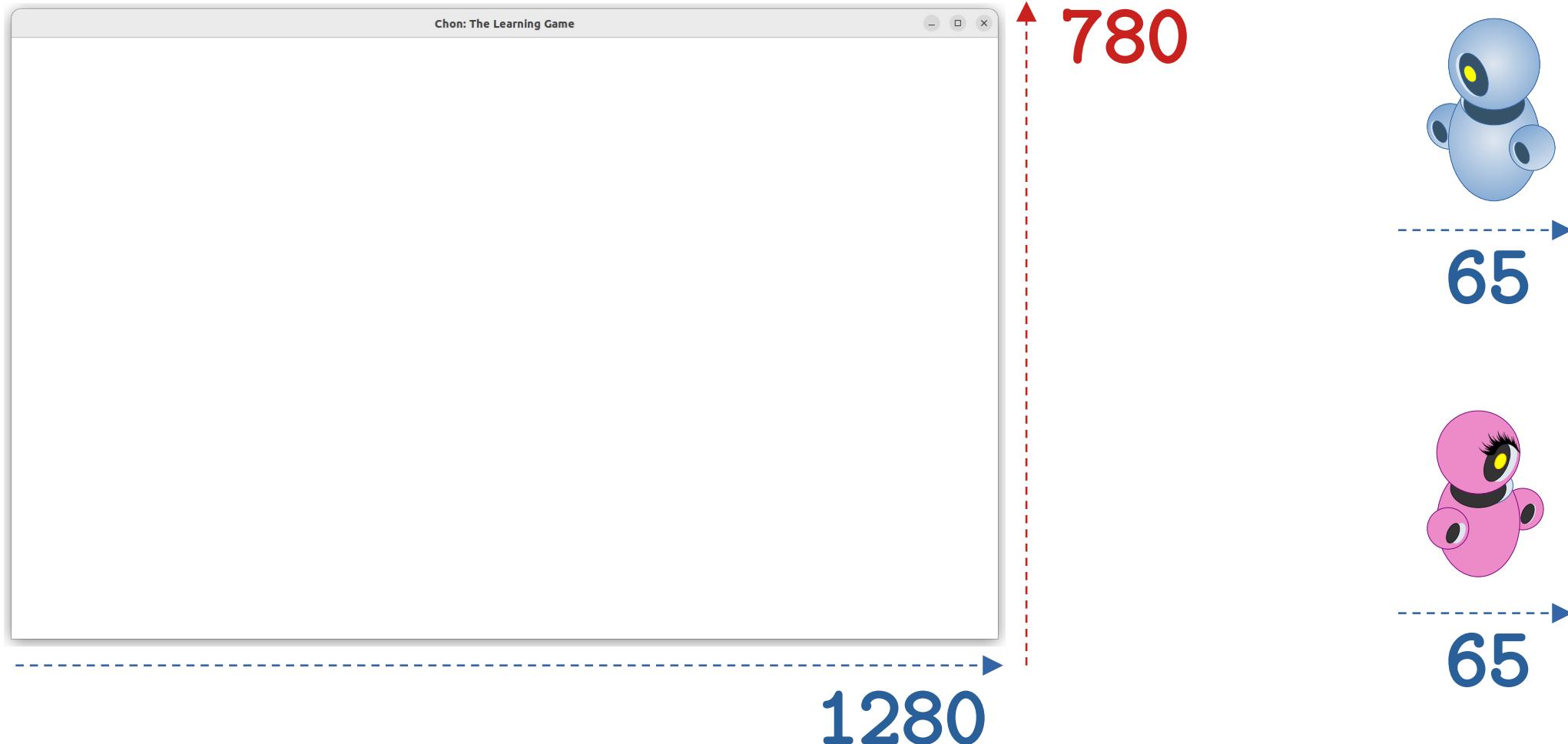
# Object's Dimension: Width and Height



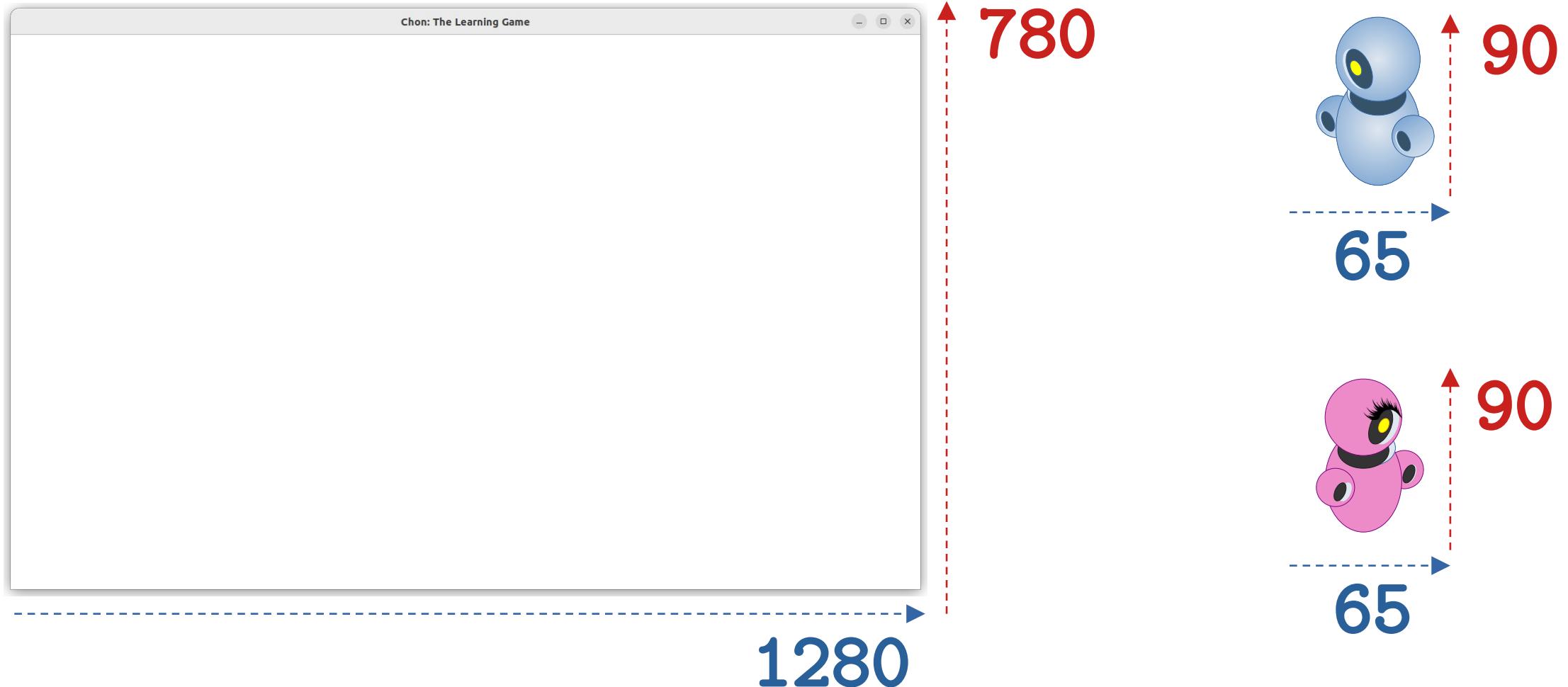
# Object's Dimension: Width and Height



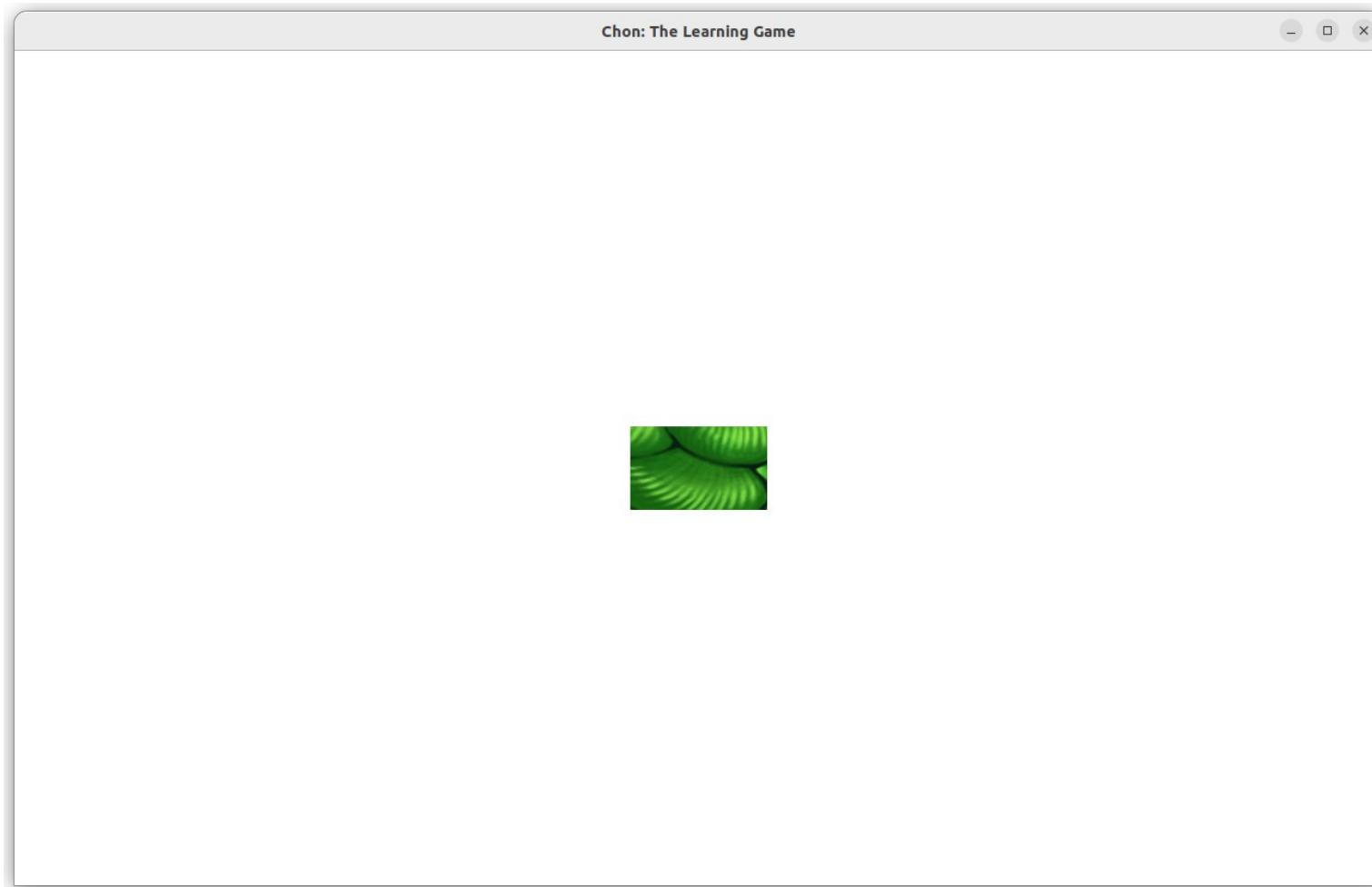
# Object's Dimension: Width and Height



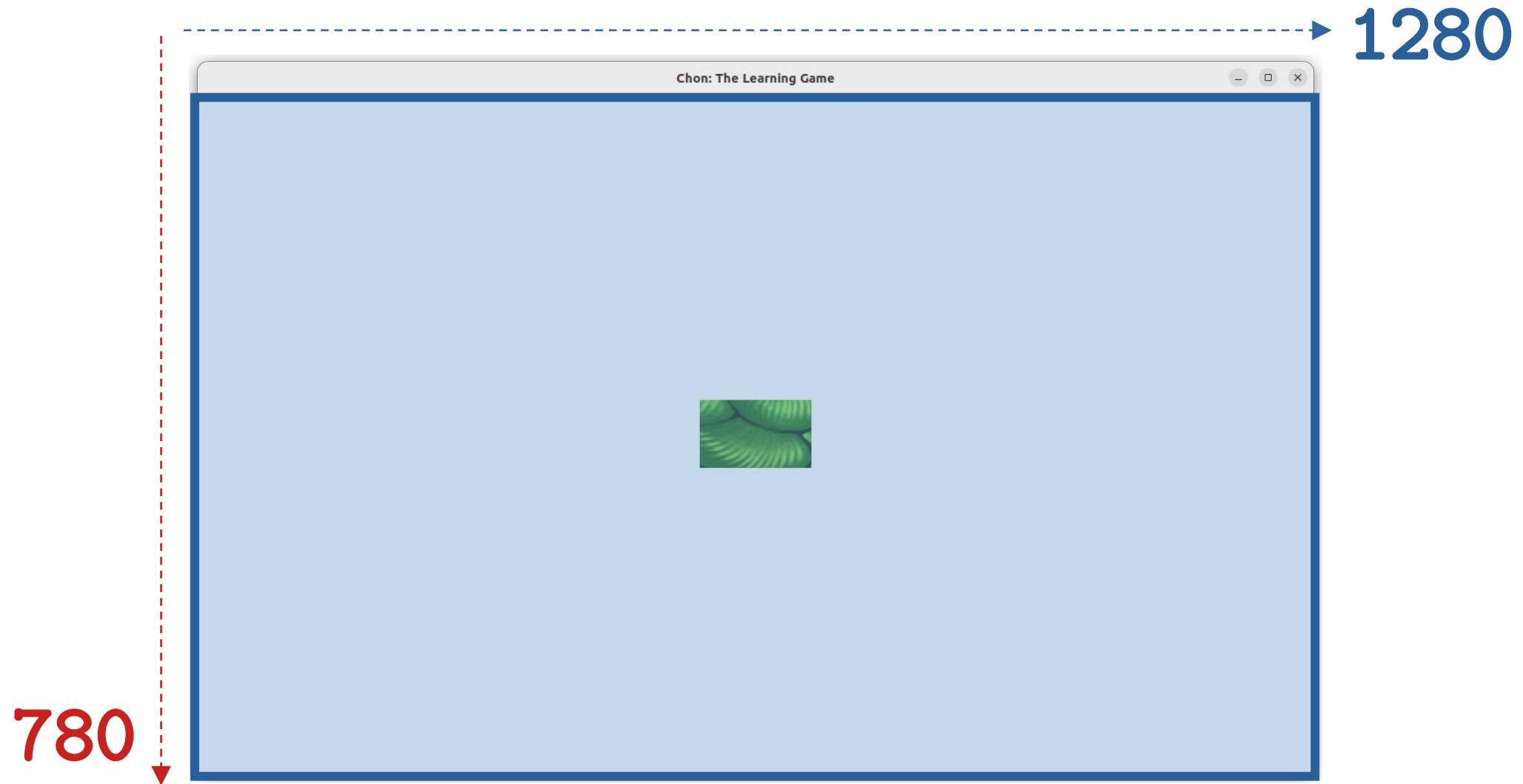
# Object's Dimension: Width and Height



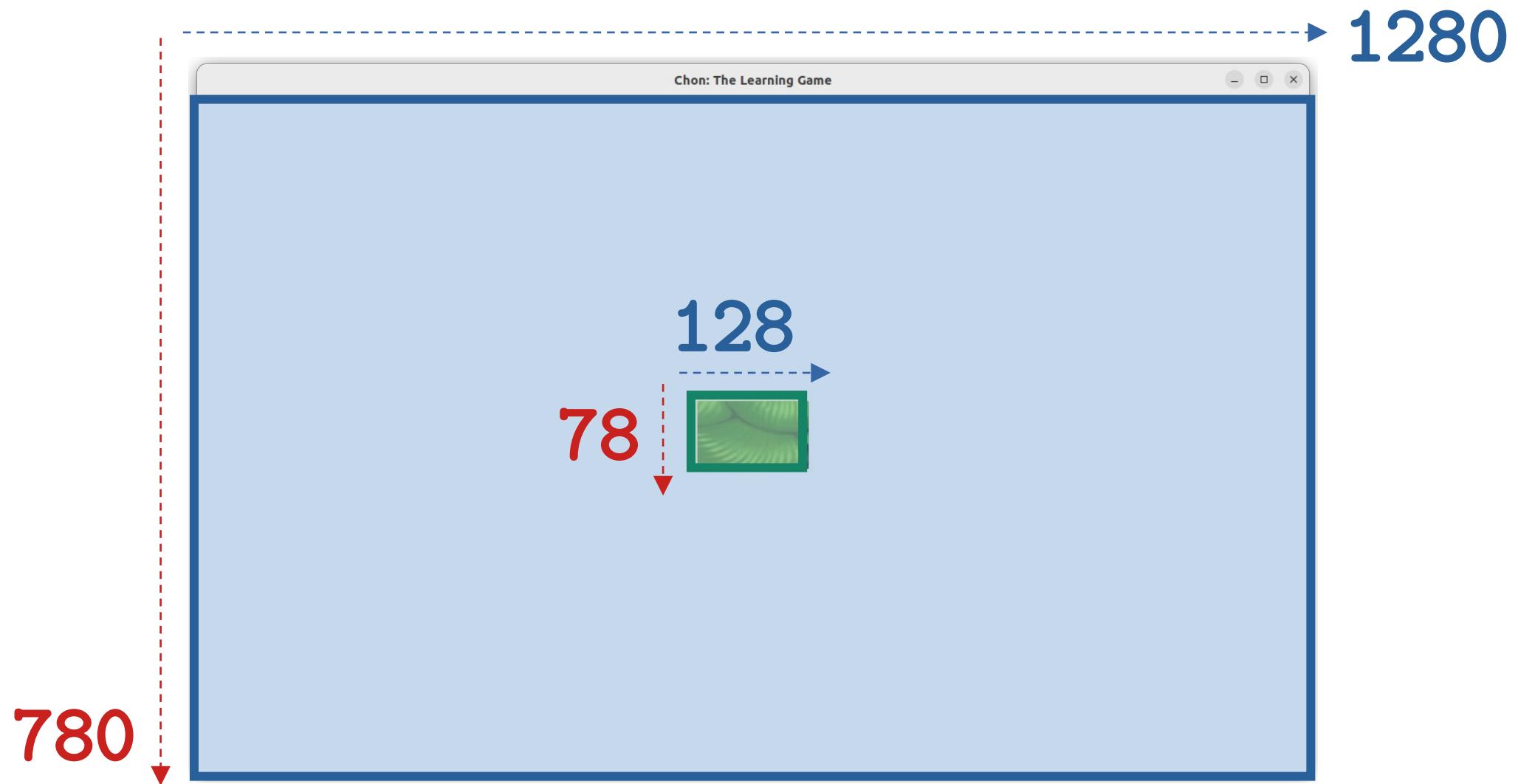
# Canvas Example



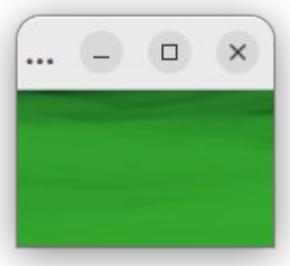
# Canvas Example



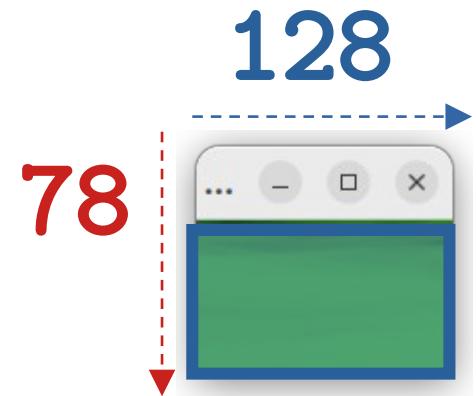
# Canvas Example



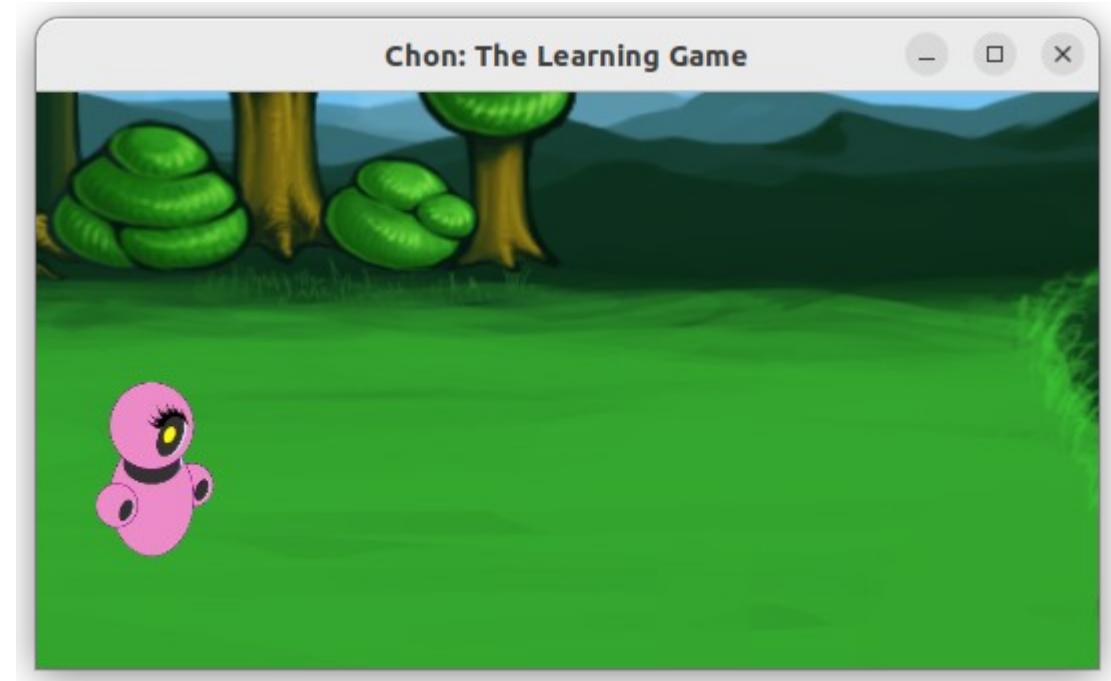
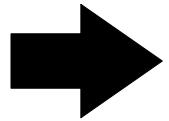
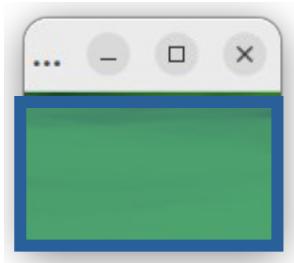
# Scene Example



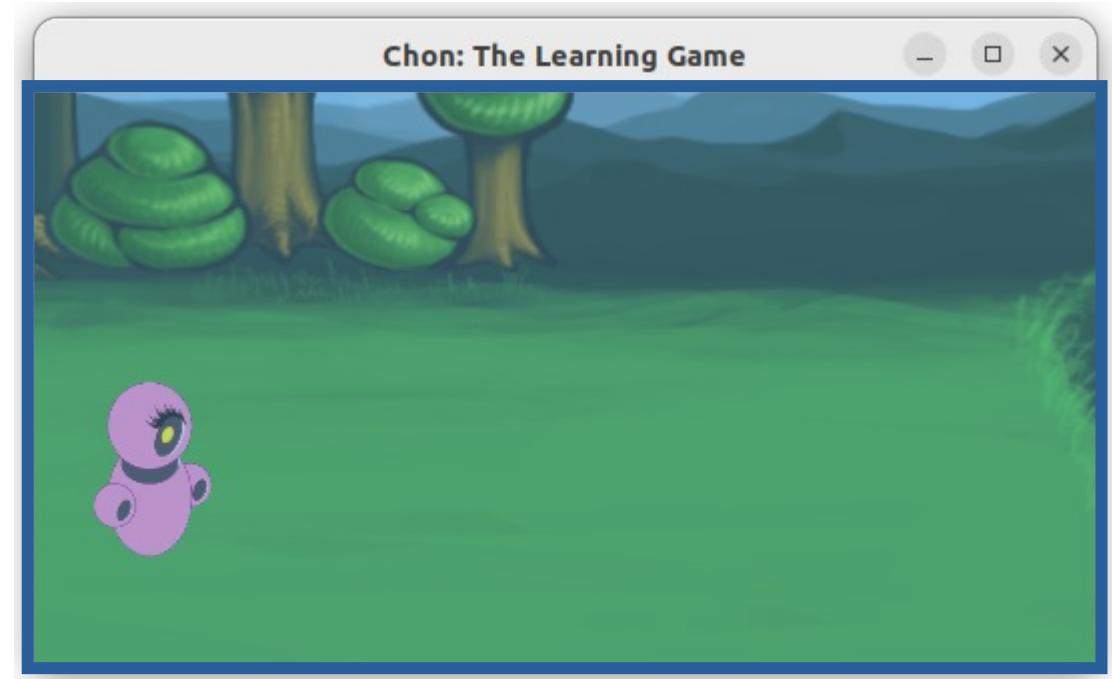
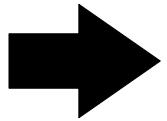
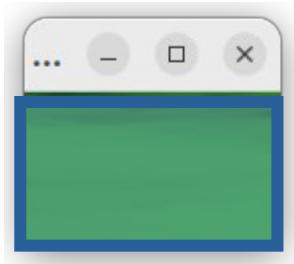
# Scene Example



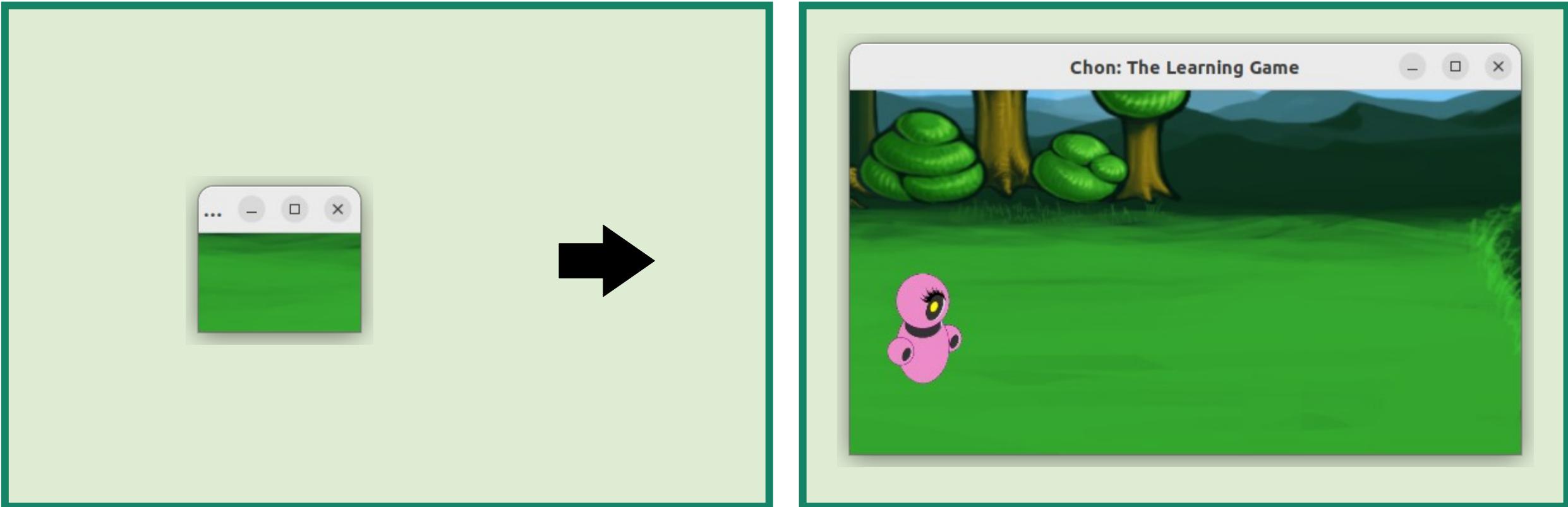
# Scene Example



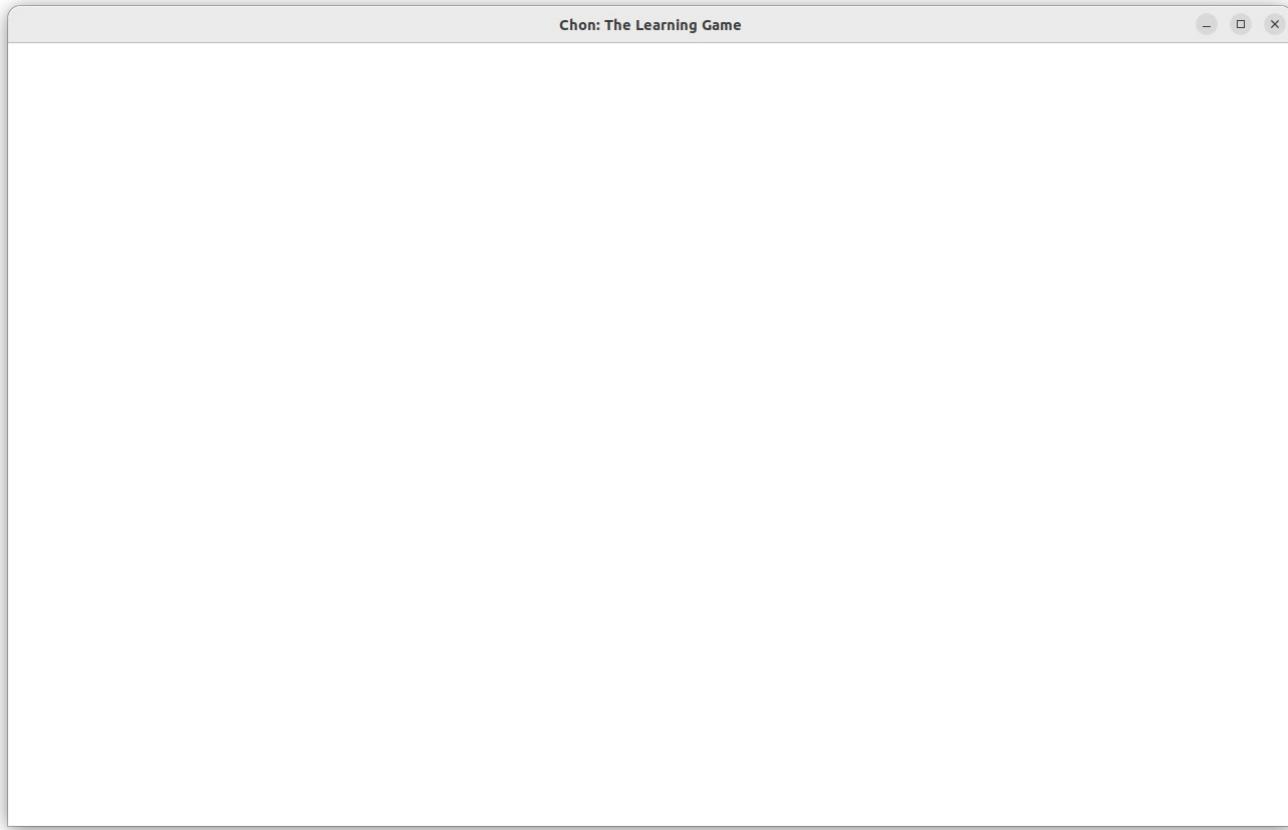
# Scene Example



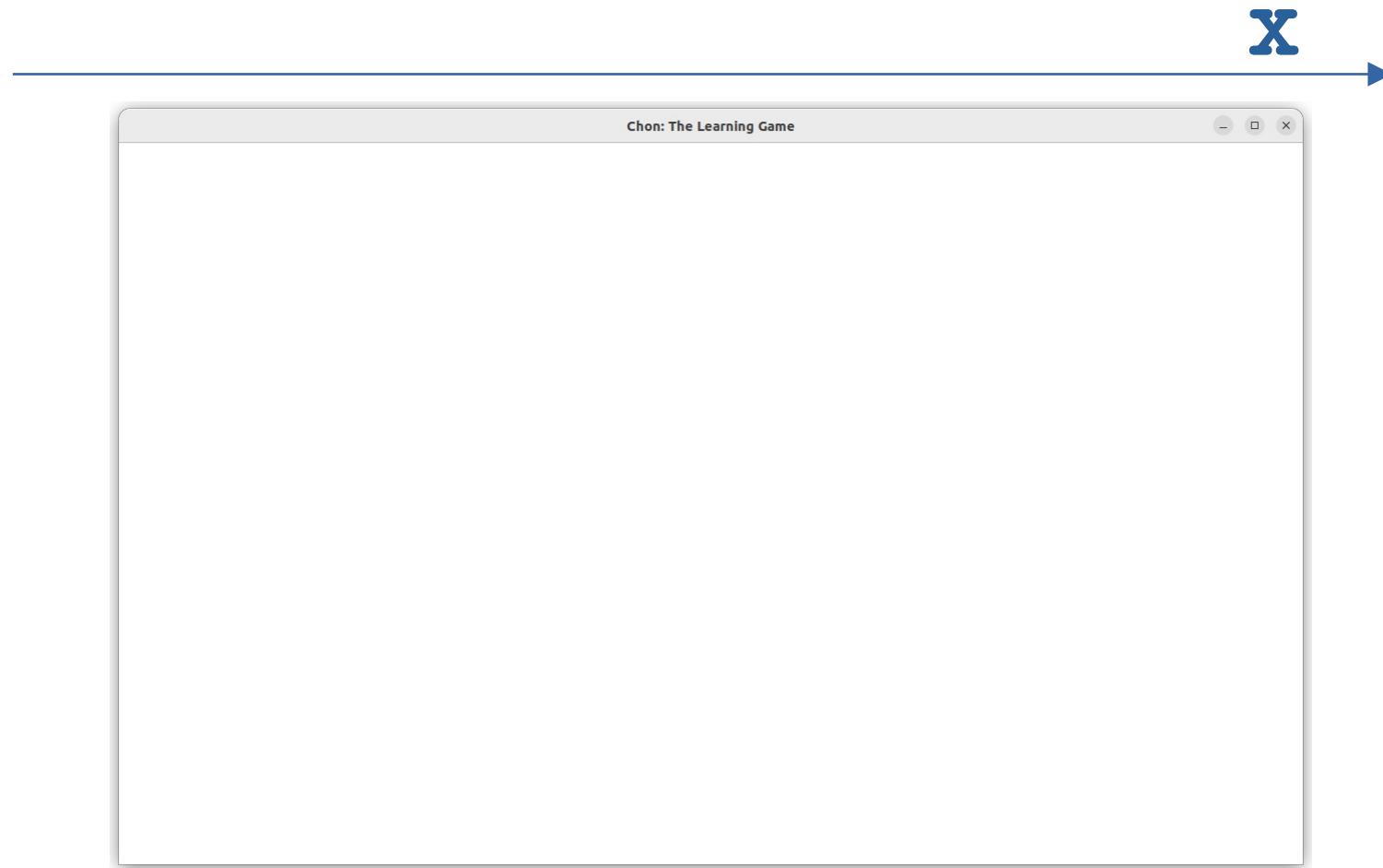
# Scene Example



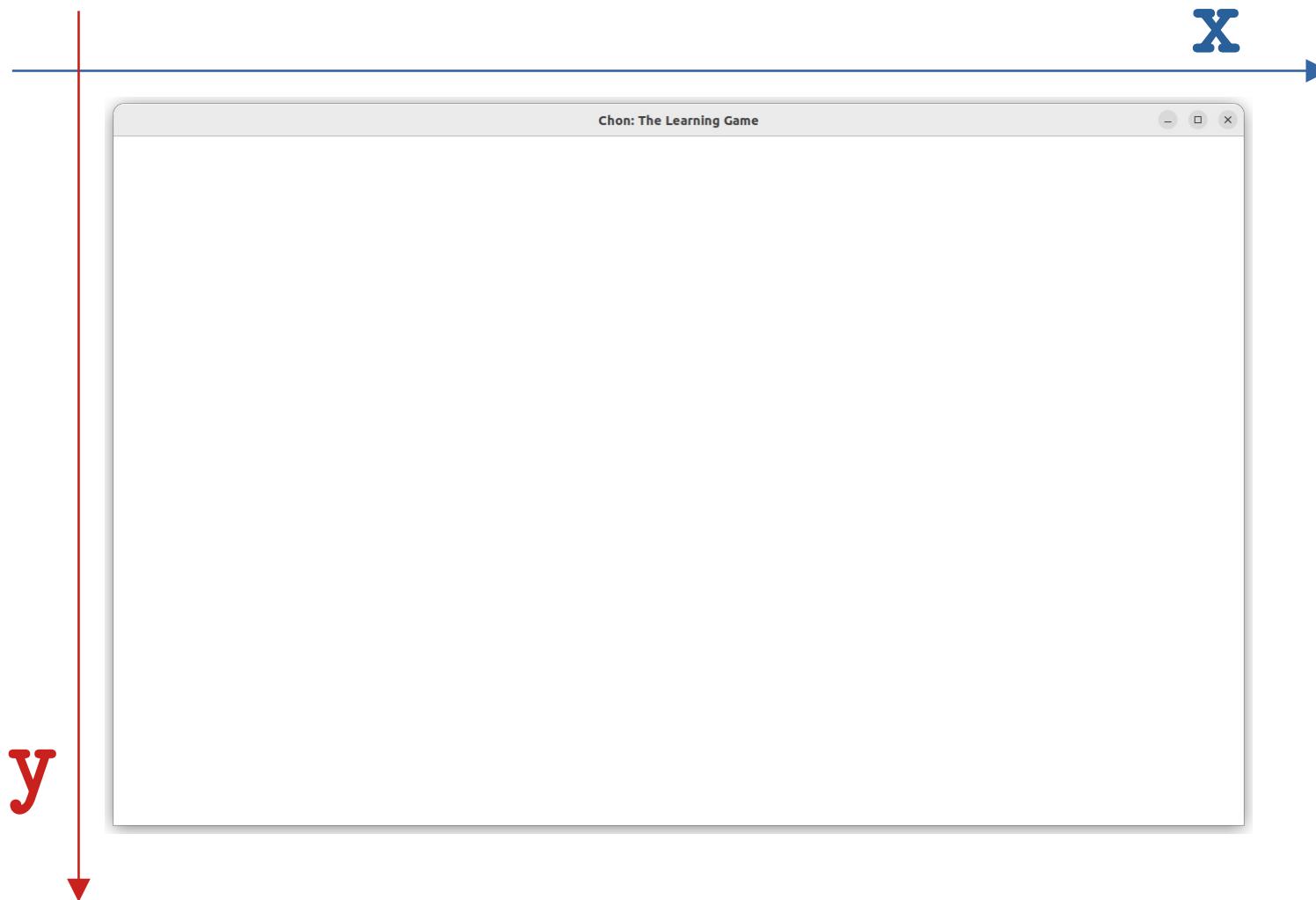
# Positioning System



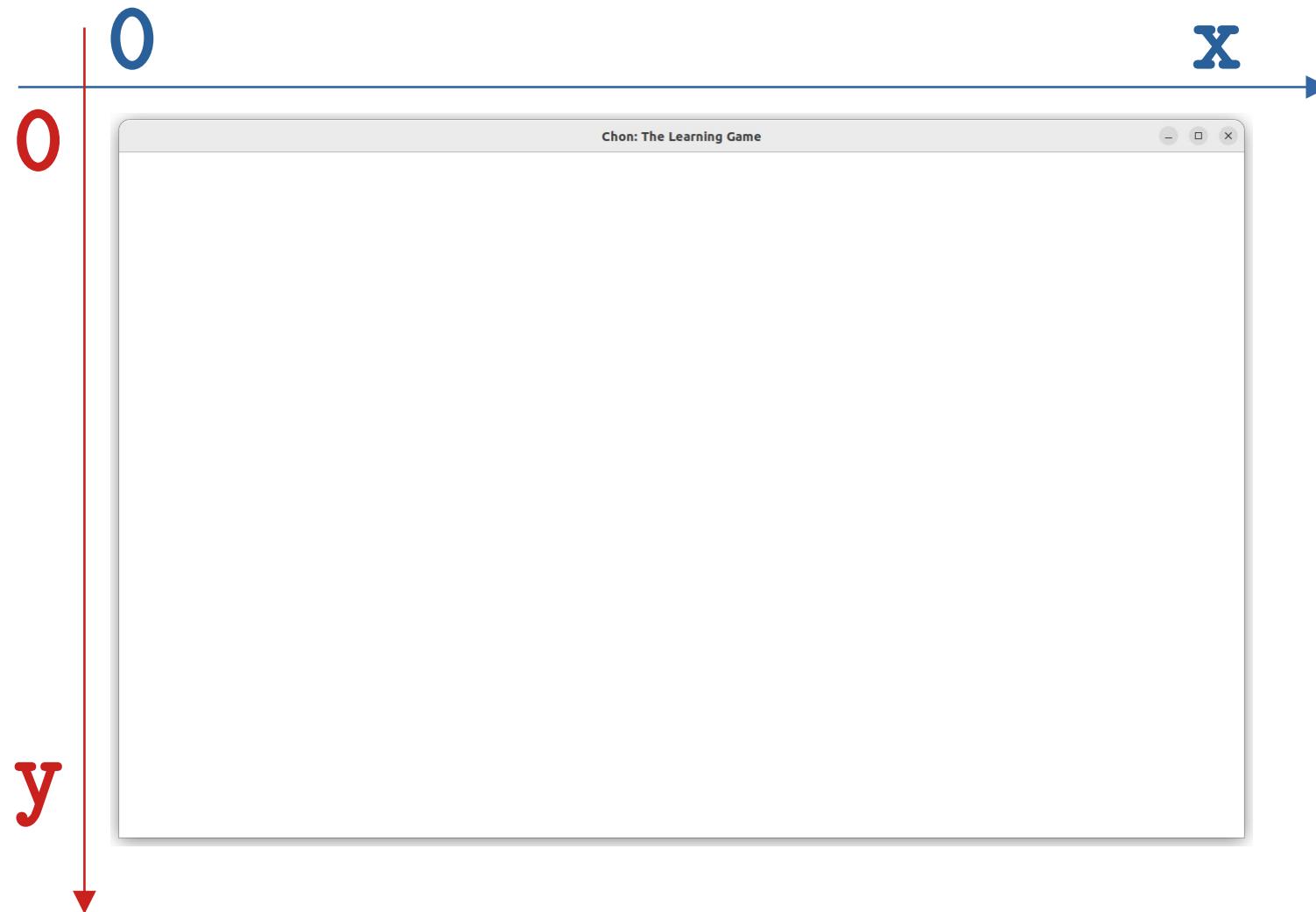
# Positioning System



# Positioning System



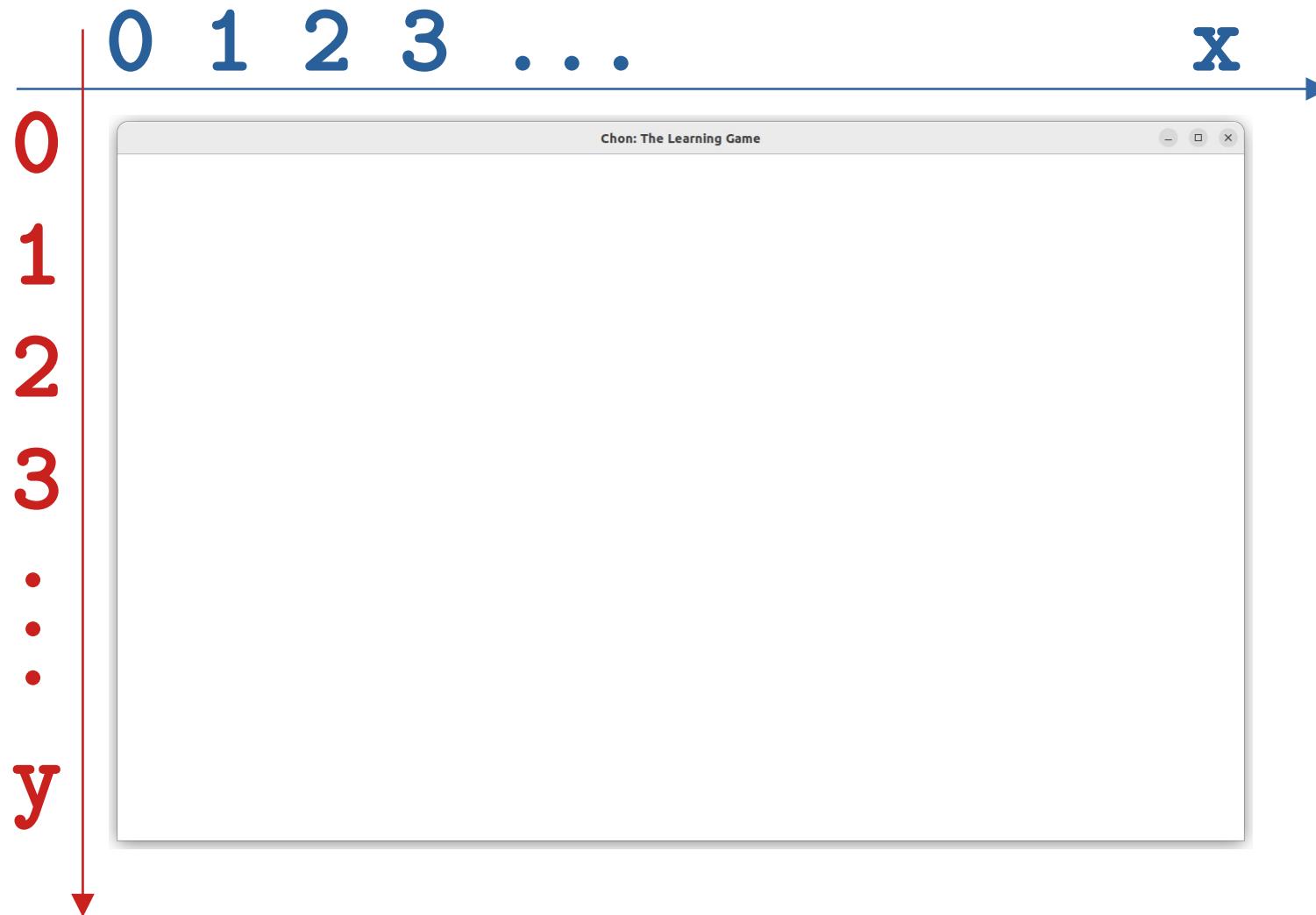
# Positioning System



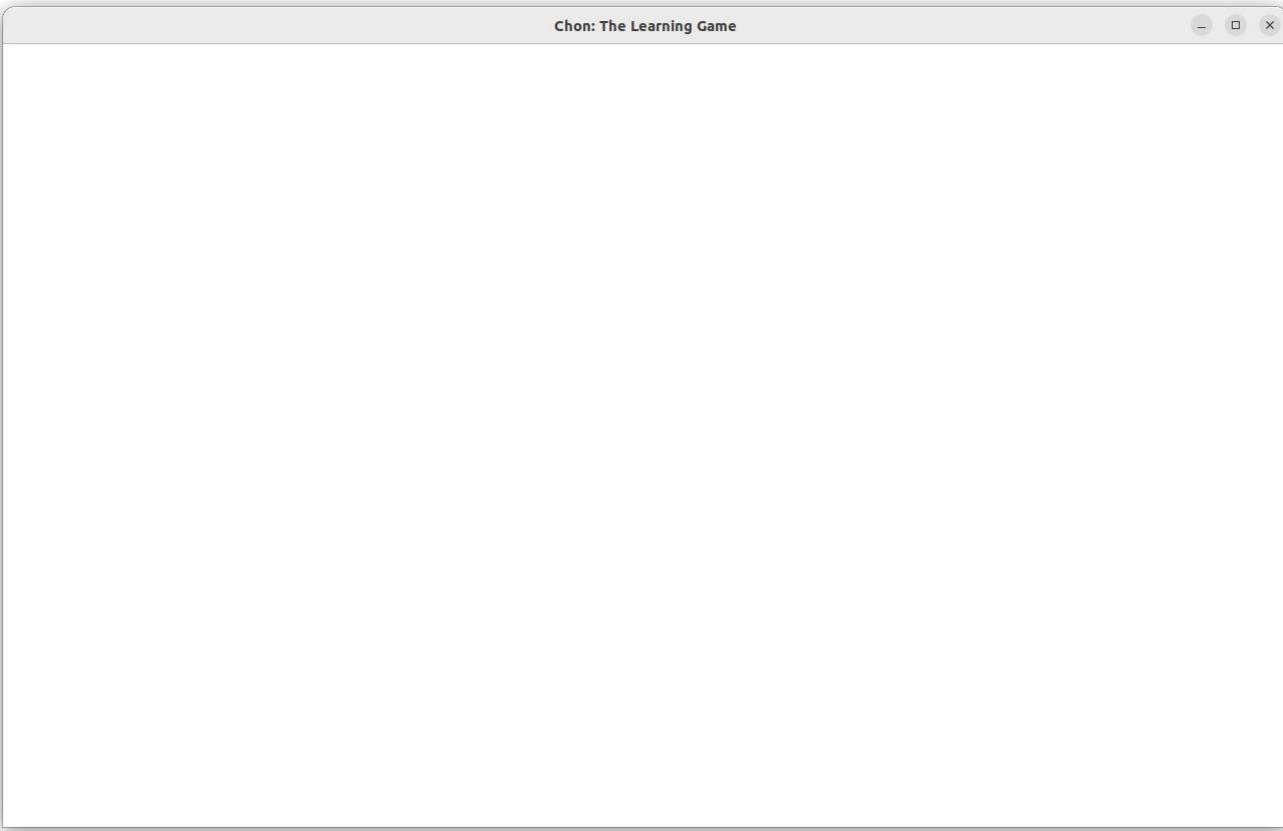
# Positioning System



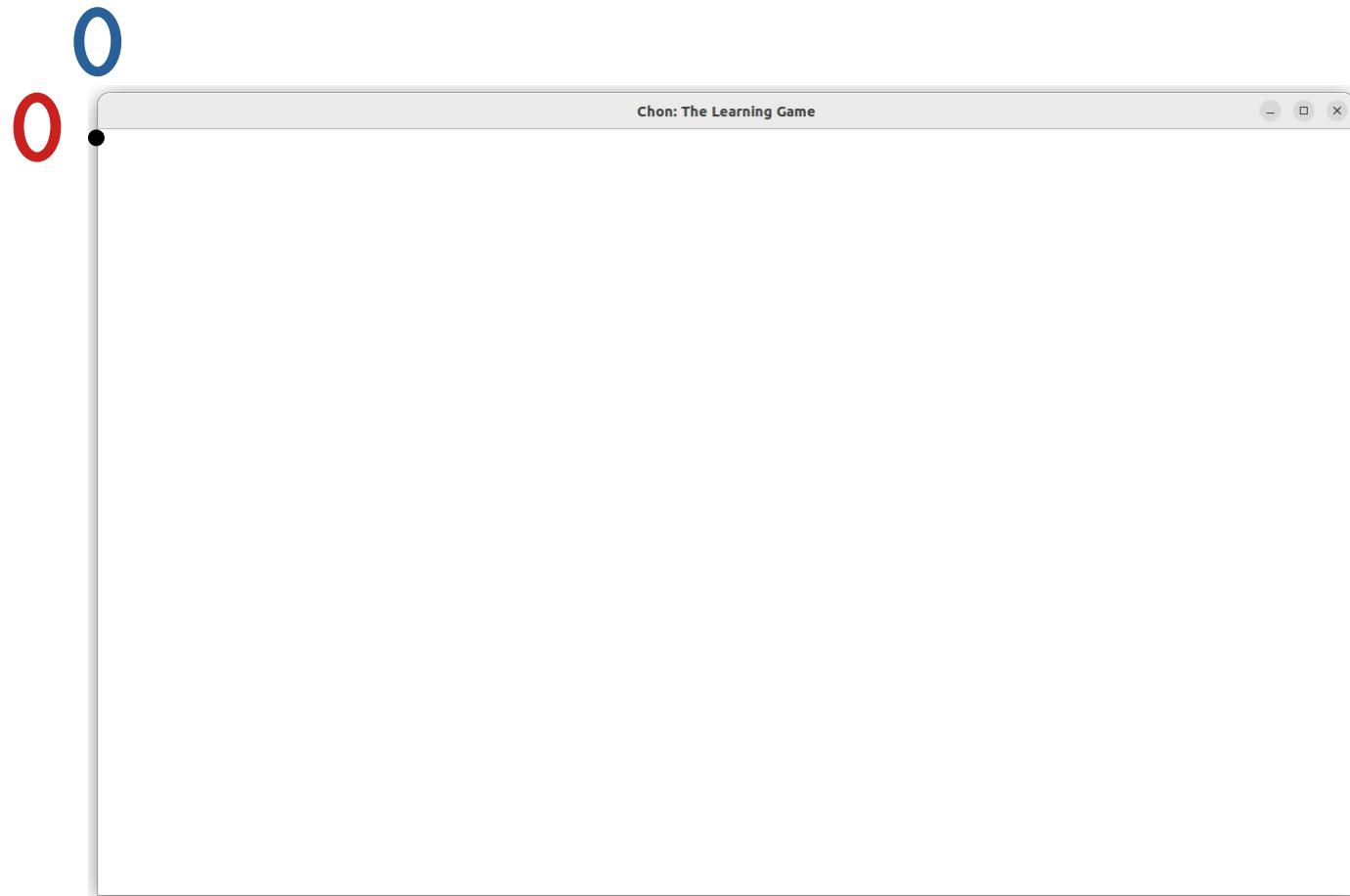
# Positioning System



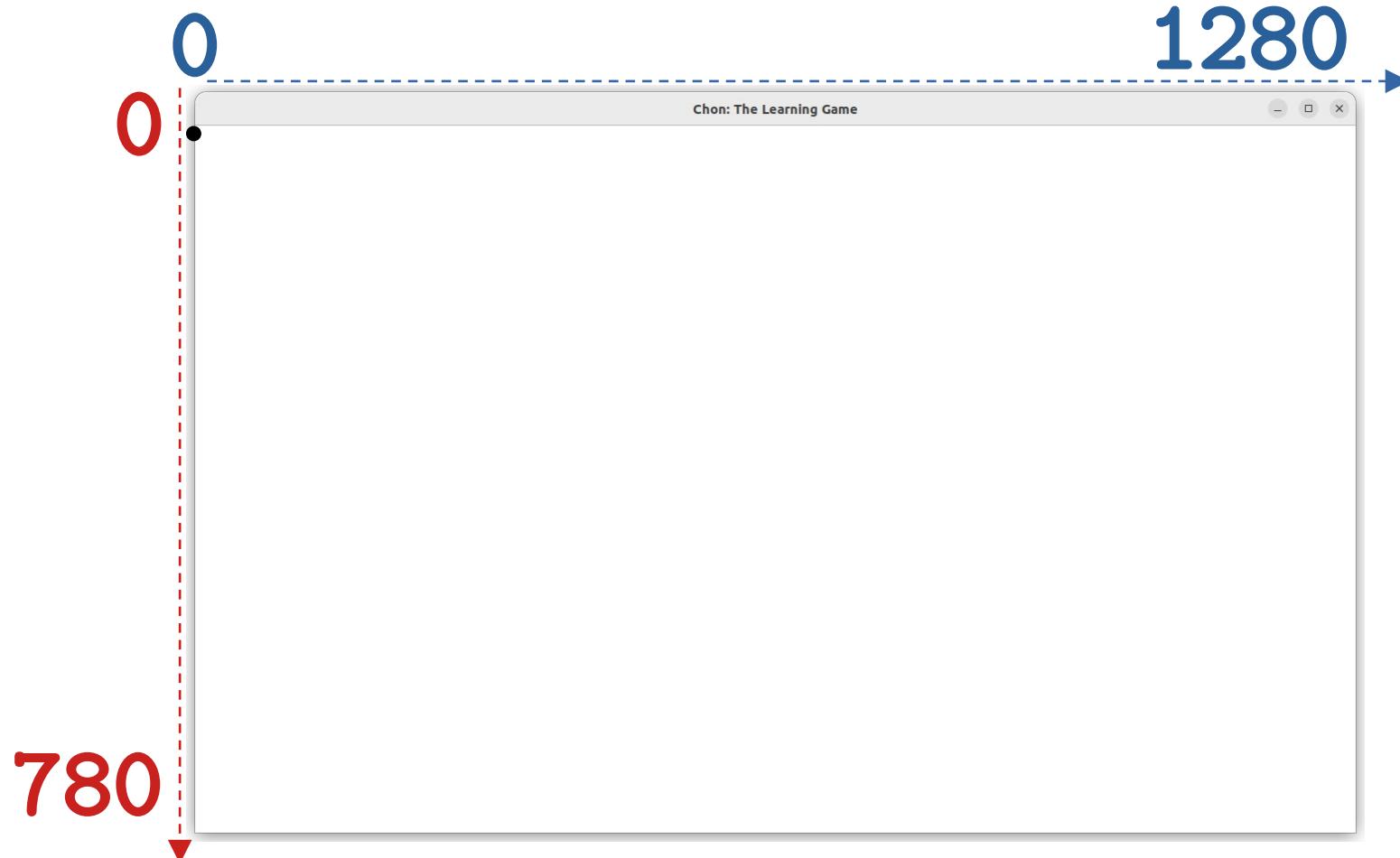
# Drawing the Background



# Drawing the Background



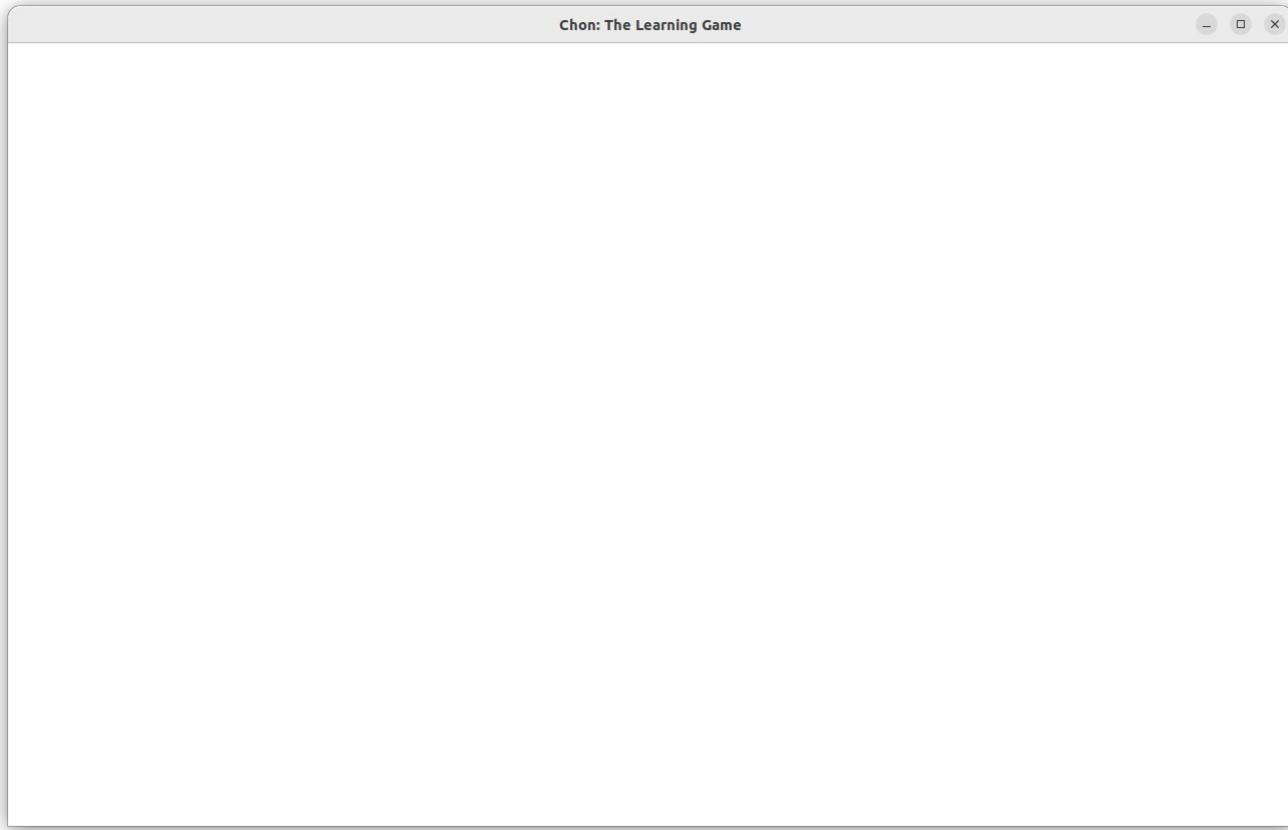
# Drawing the Background



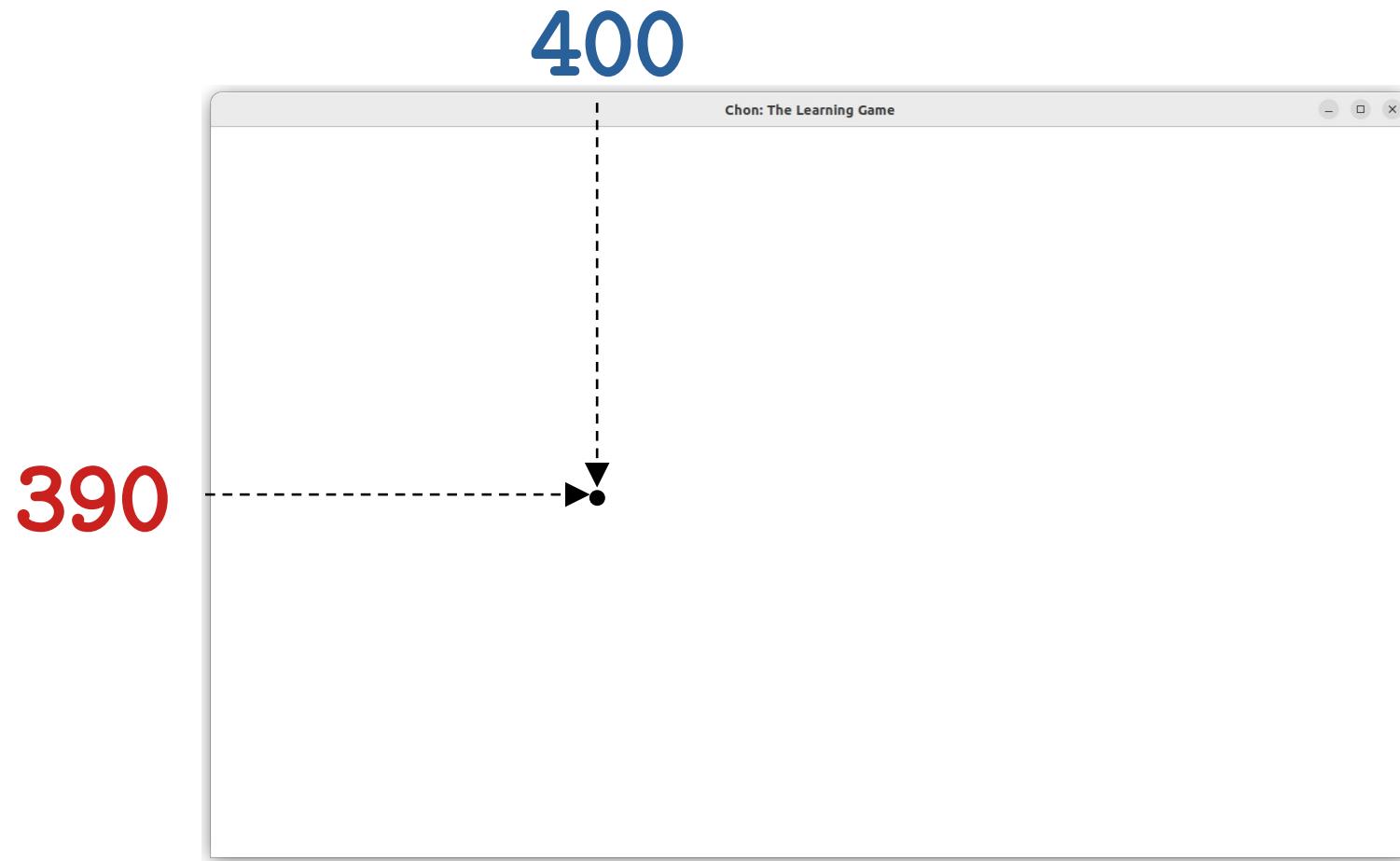
# Drawing the Background



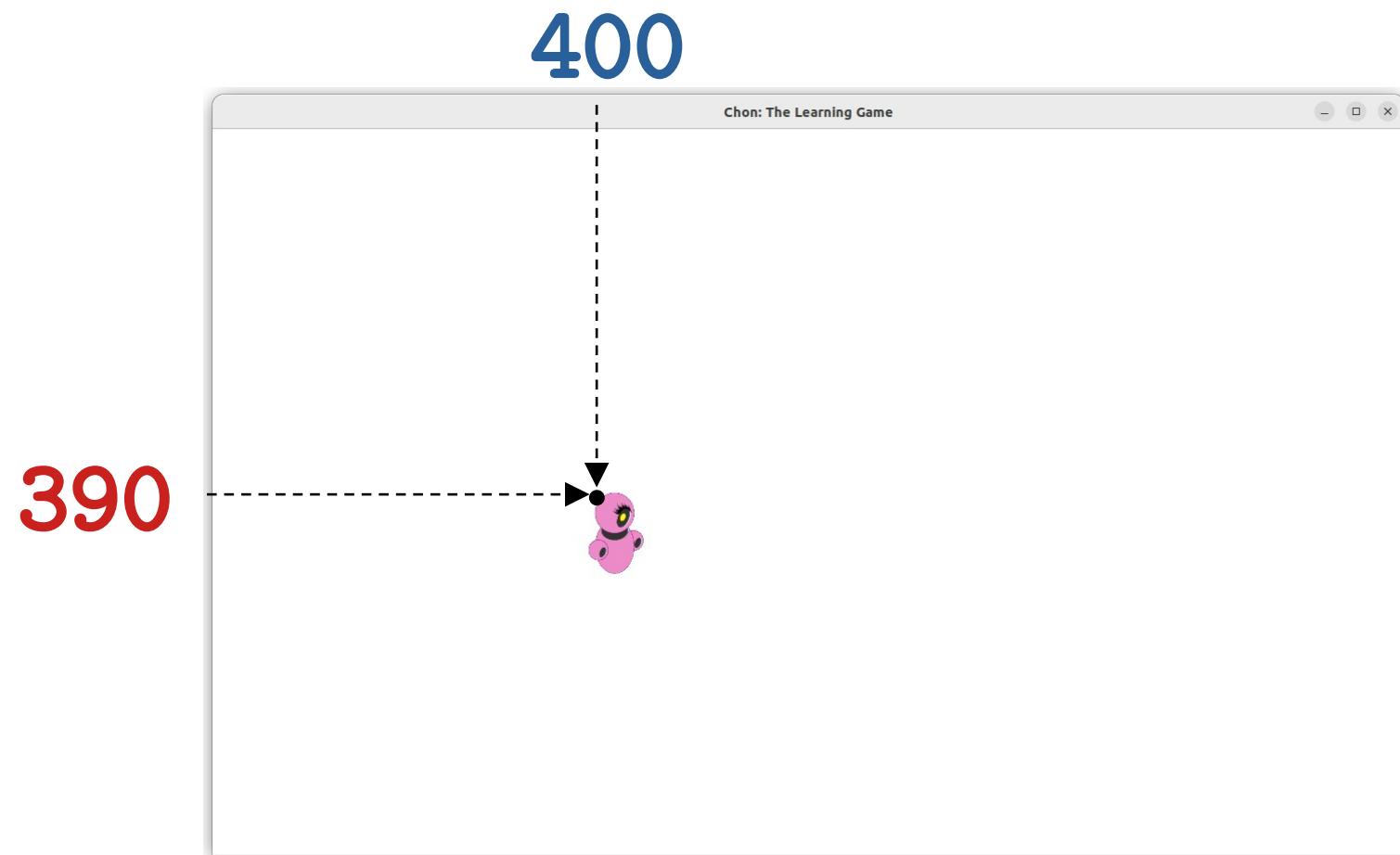
# Drawing the Characters



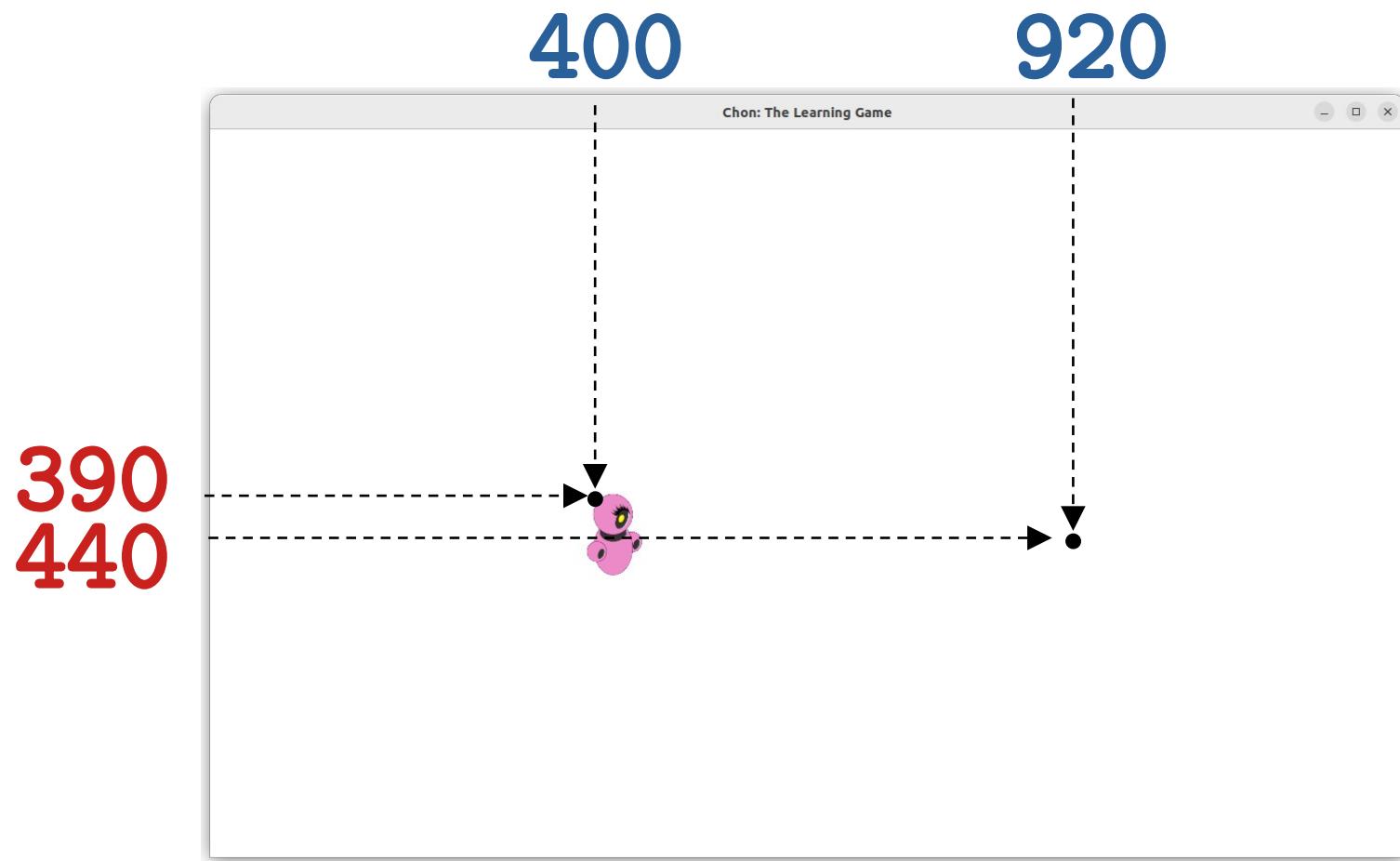
# Drawing the Characters



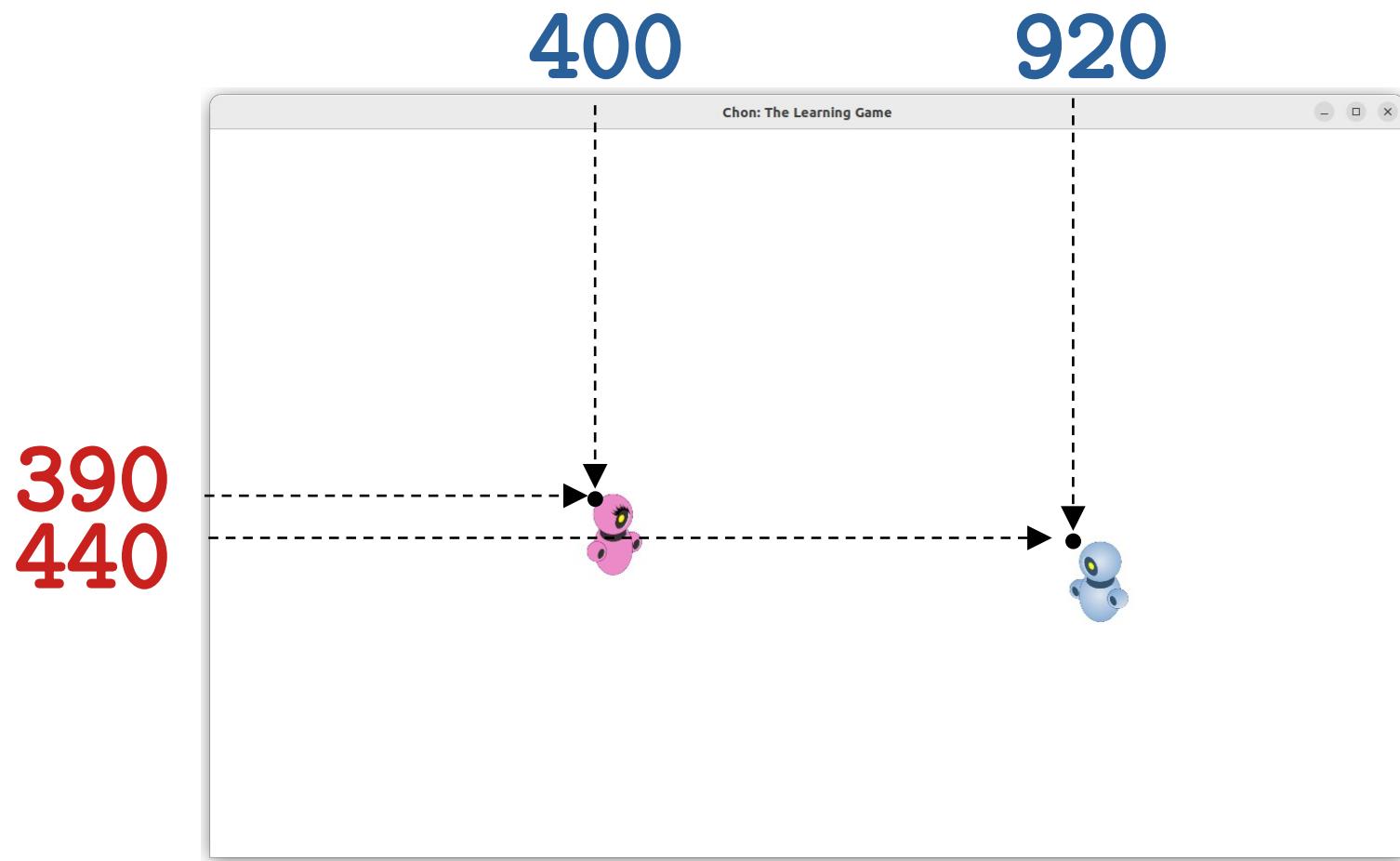
# Drawing the Characters



# Drawing the Characters



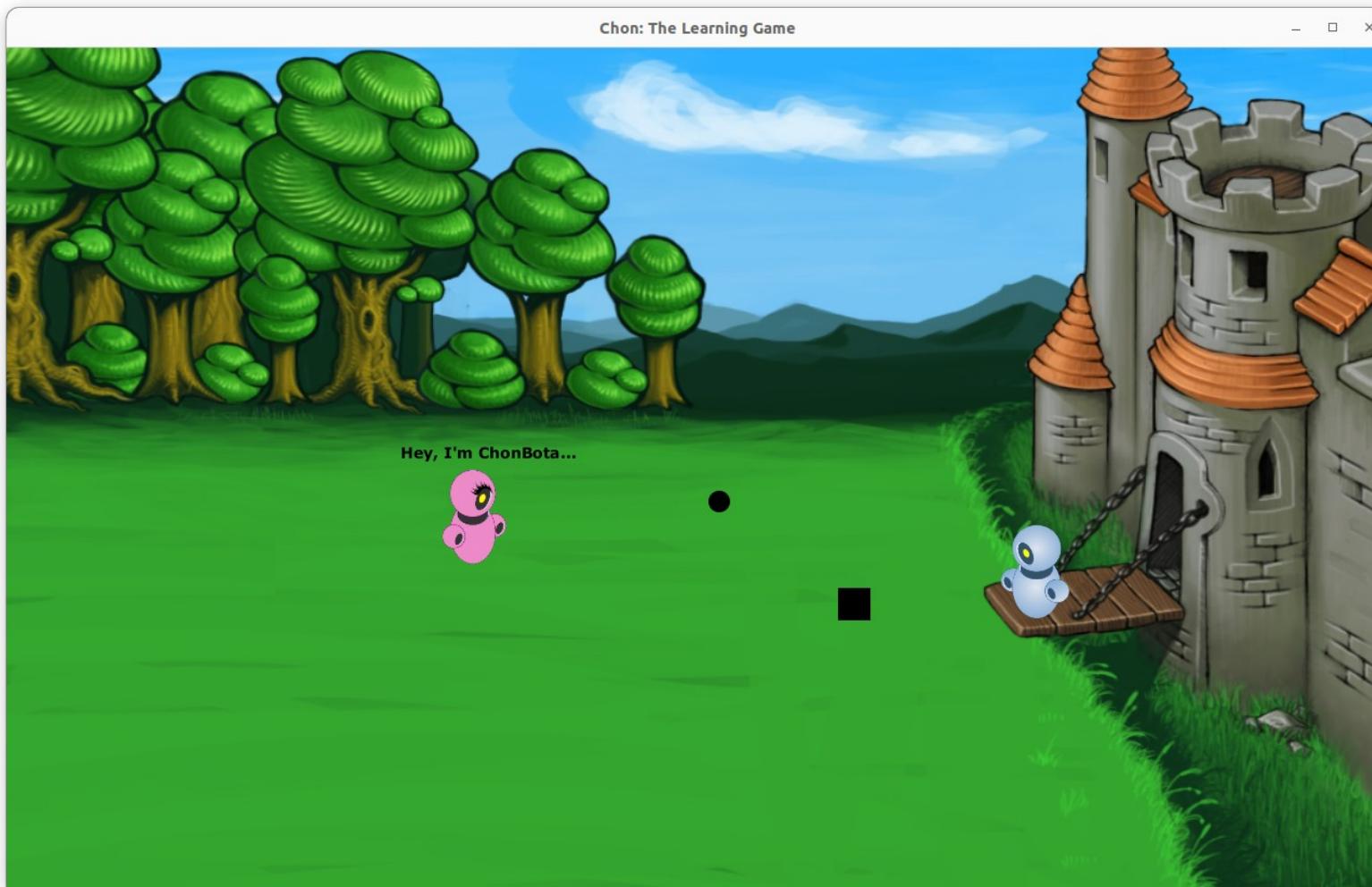
# Drawing the Characters



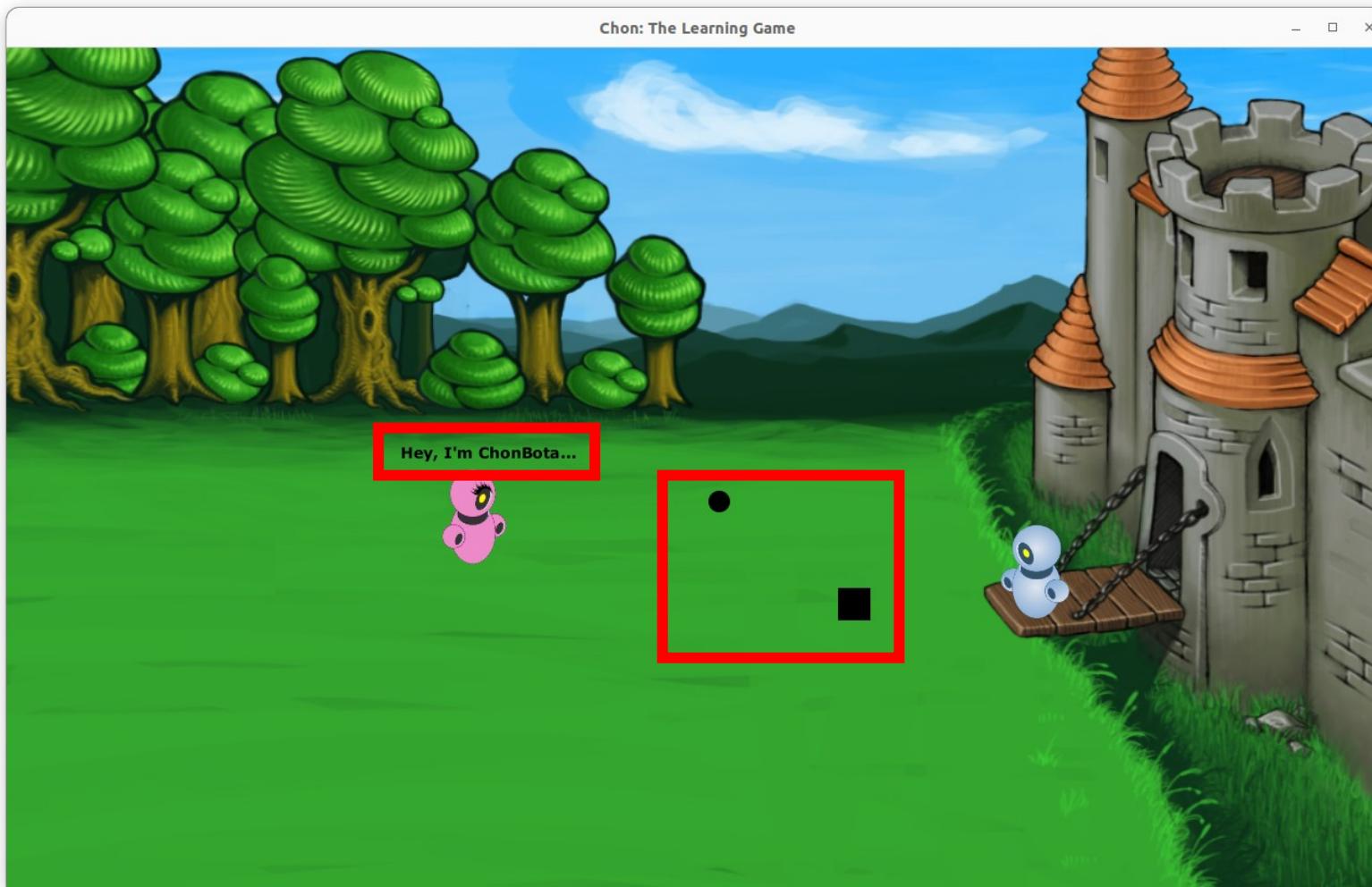
# Drawing the Characters



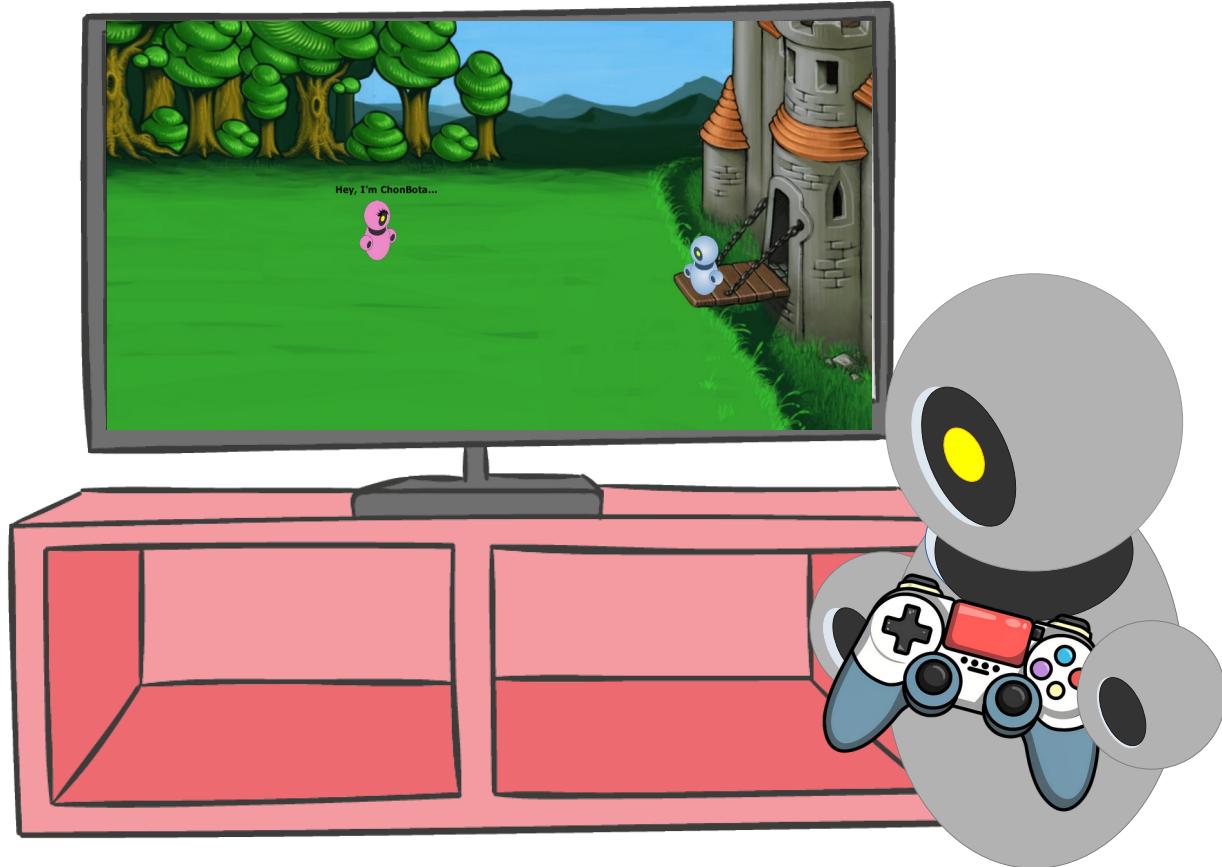
# Drawing Other Objects



# Drawing Other Objects



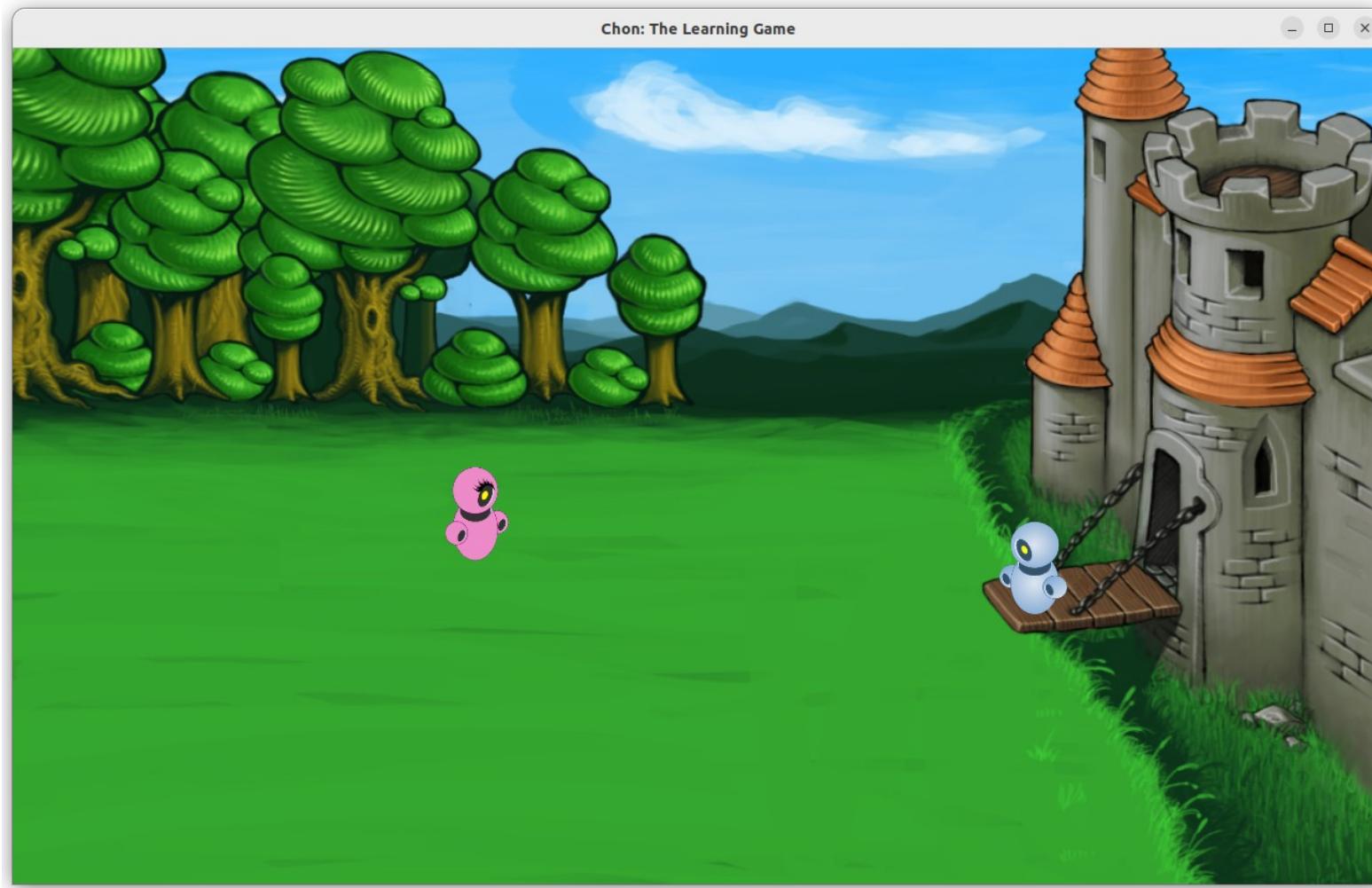
# MOVING OBJECTS IN THE CANVAS



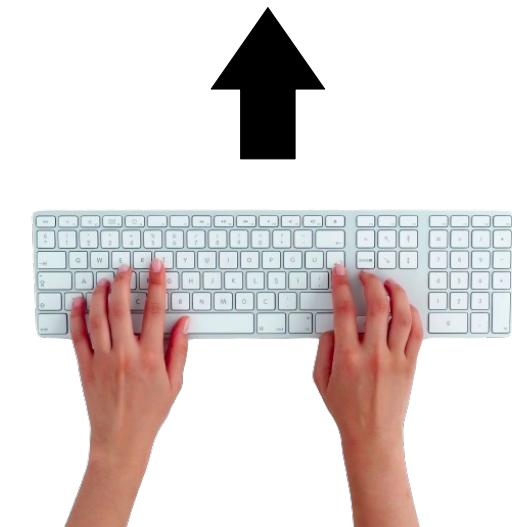
# Scene's Events

In JavaFX, **Events** represent interactions or changes that happen while running a JavaFX application, such as **user inputs** (mouse clicks, key presses), **window actions** (resizing, closing), or **internal changes** (animation updates).

# Events



# Events



# Events



- Pressed: UP  
Released: UP
- Pressed: DOWN  
Released: DOWN
- Pressed: RIGHT  
Released: RIGHT
- Pressed: SPACE  
Released: SPACE
- Pressed: UP  
Released: UP

□



# Animation

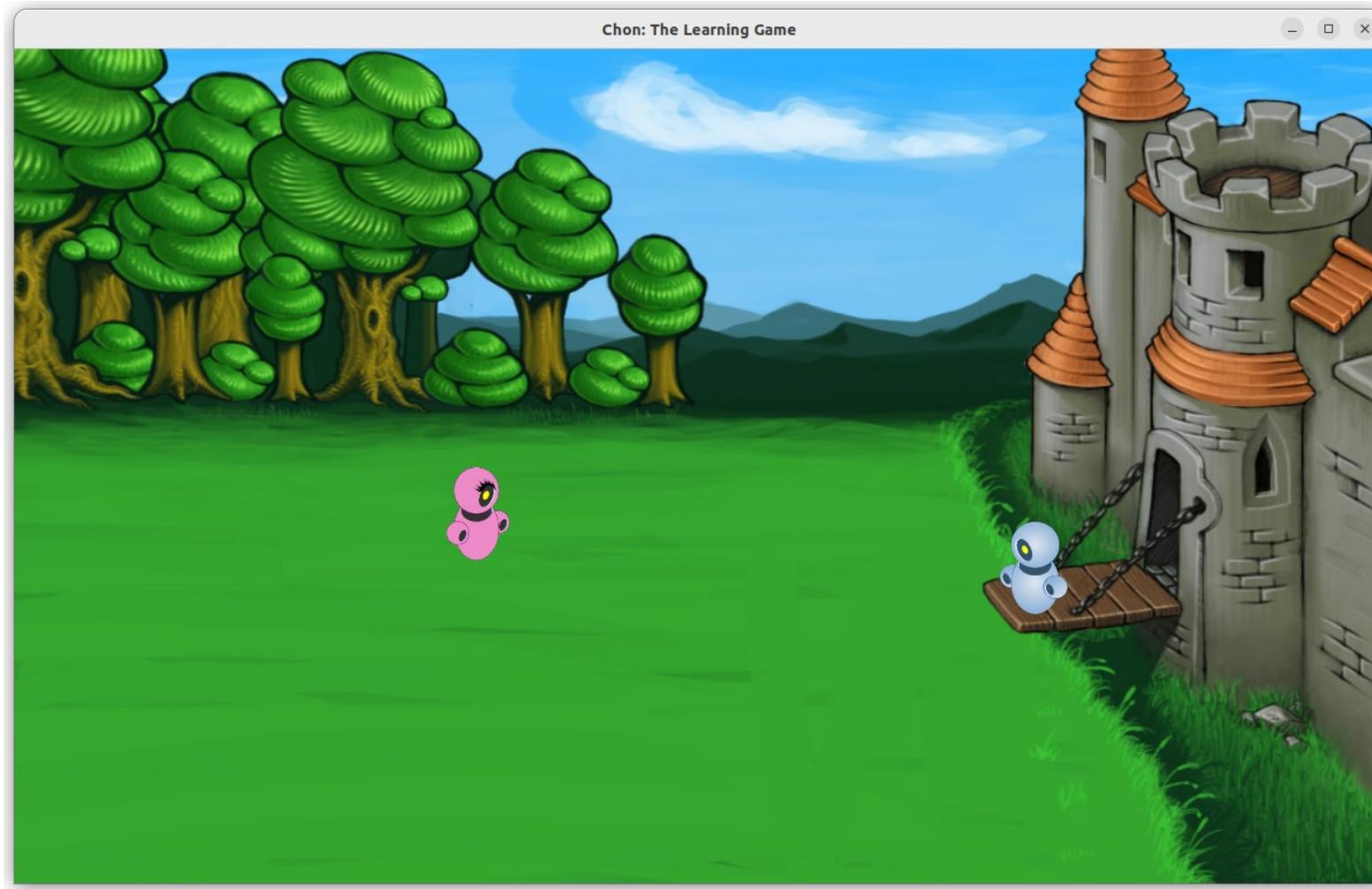
The **AnimationTimer** allows the creation of animations by repeatedly executing a code block at each frame.

# Animation

The **AnimationTimer** allows the creation of animations by repeatedly executing a code block at each frame.

It provides a way to perform updates and render graphics, making it suitable for **creating animations, game loops, and other time-based tasks**.

# Animation



# Animation



○ Pressed: RIGHT  
Released: RIGHT  
□



# Animation



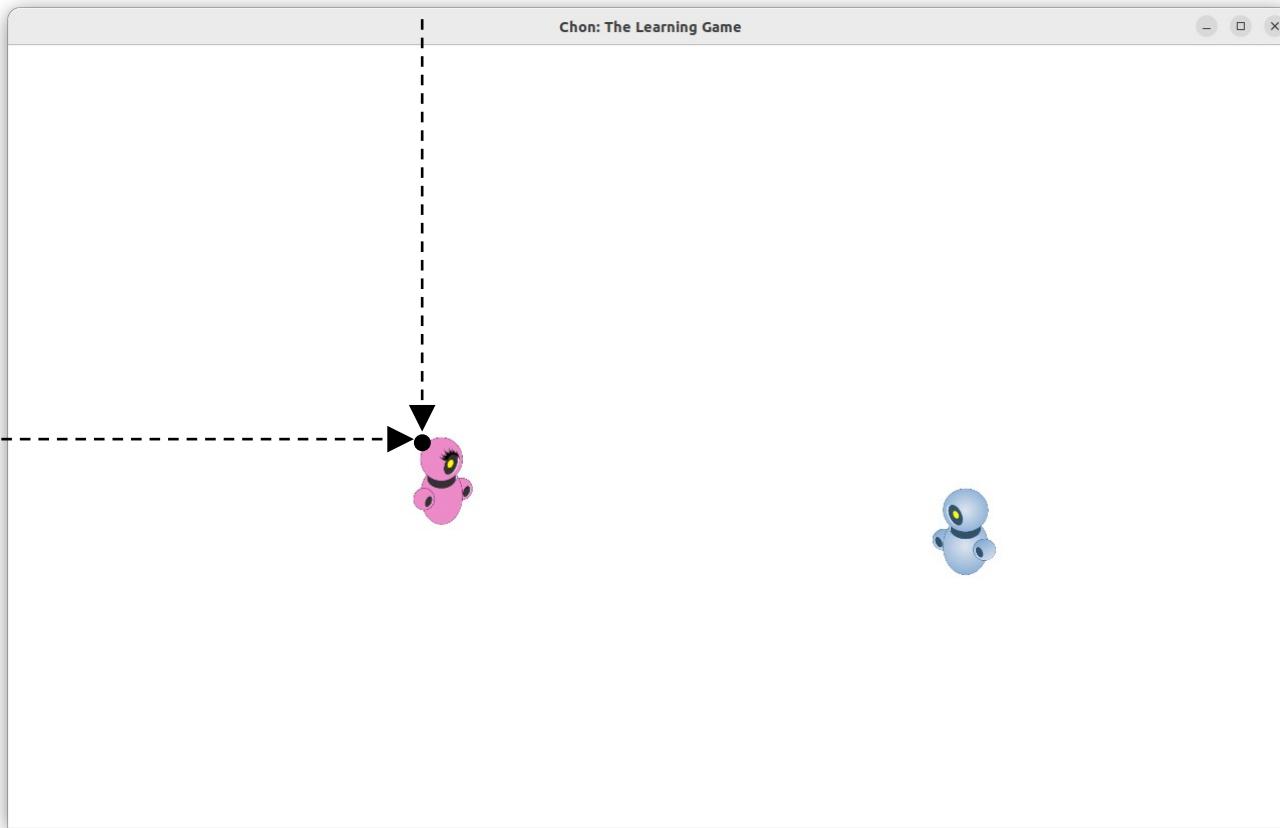
○ Pressed: RIGHT  
Released: RIGHT  
□



# The RIGHT Logic

400

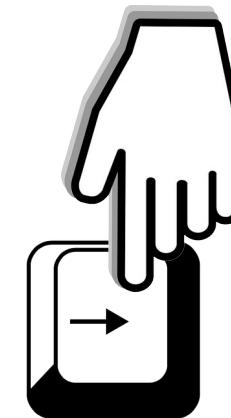
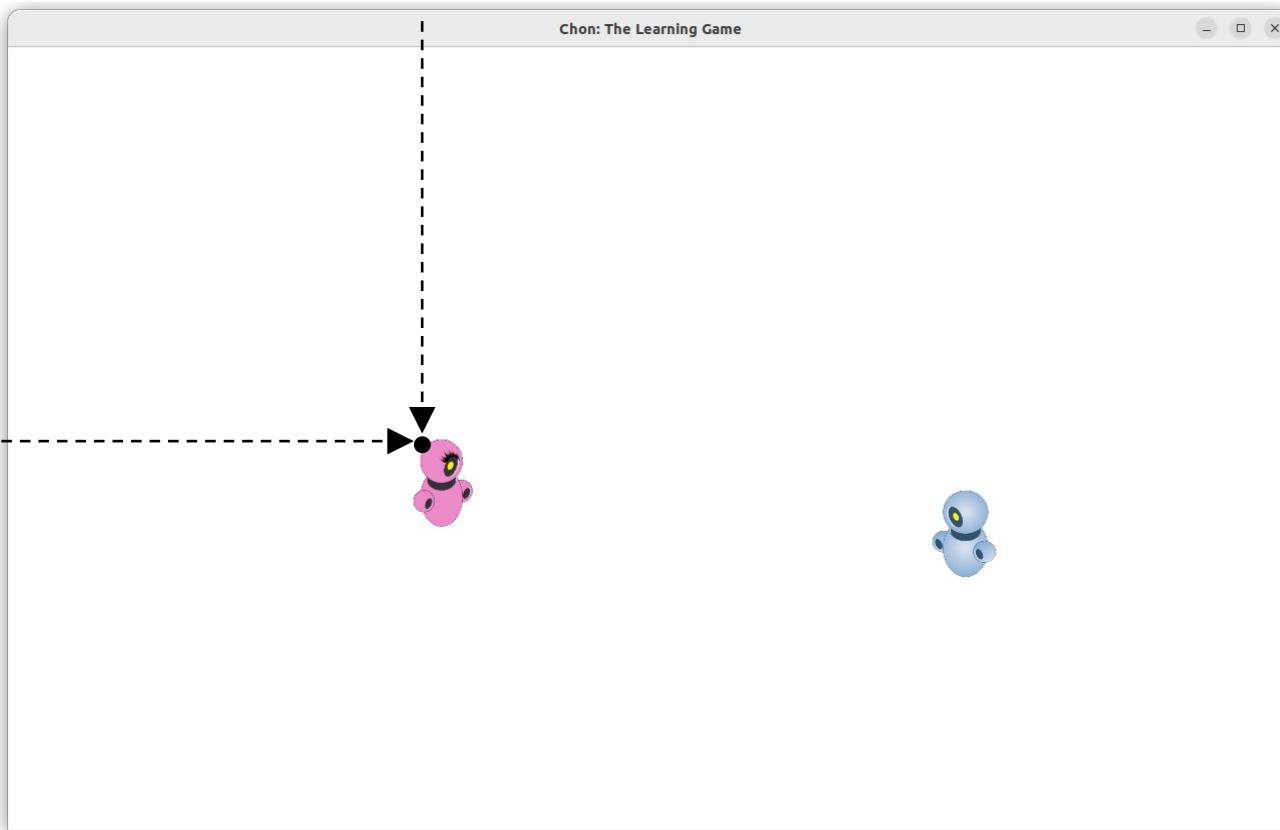
390



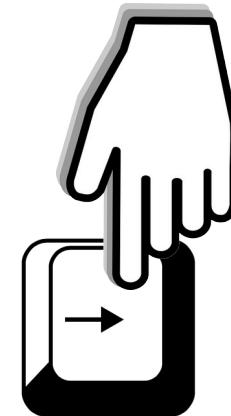
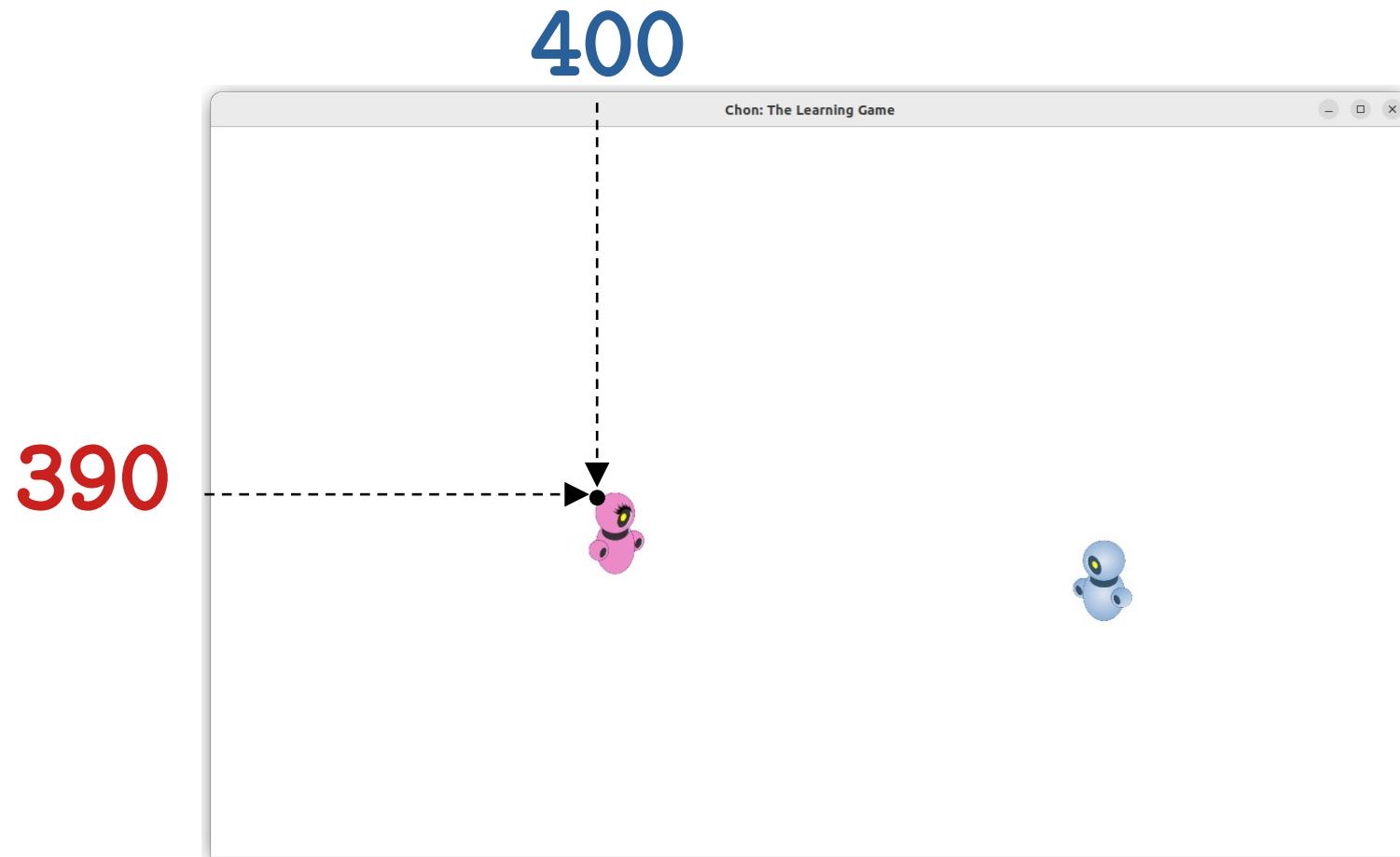
# The RIGHT Logic

400

390



# The RIGHT Logic

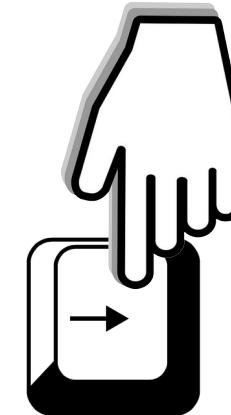
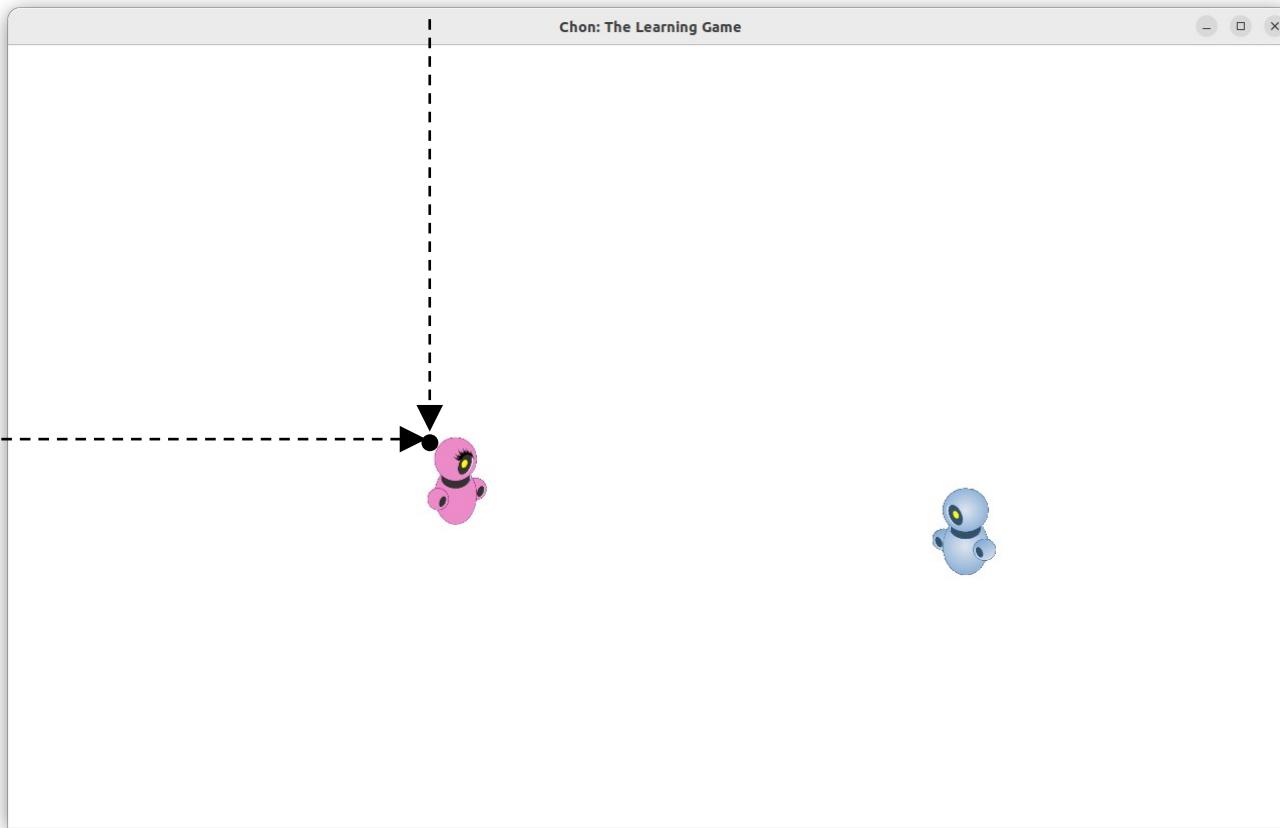


$x + \text{value}$

# The RIGHT Logic

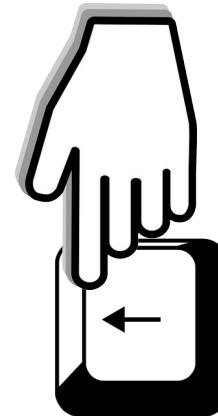
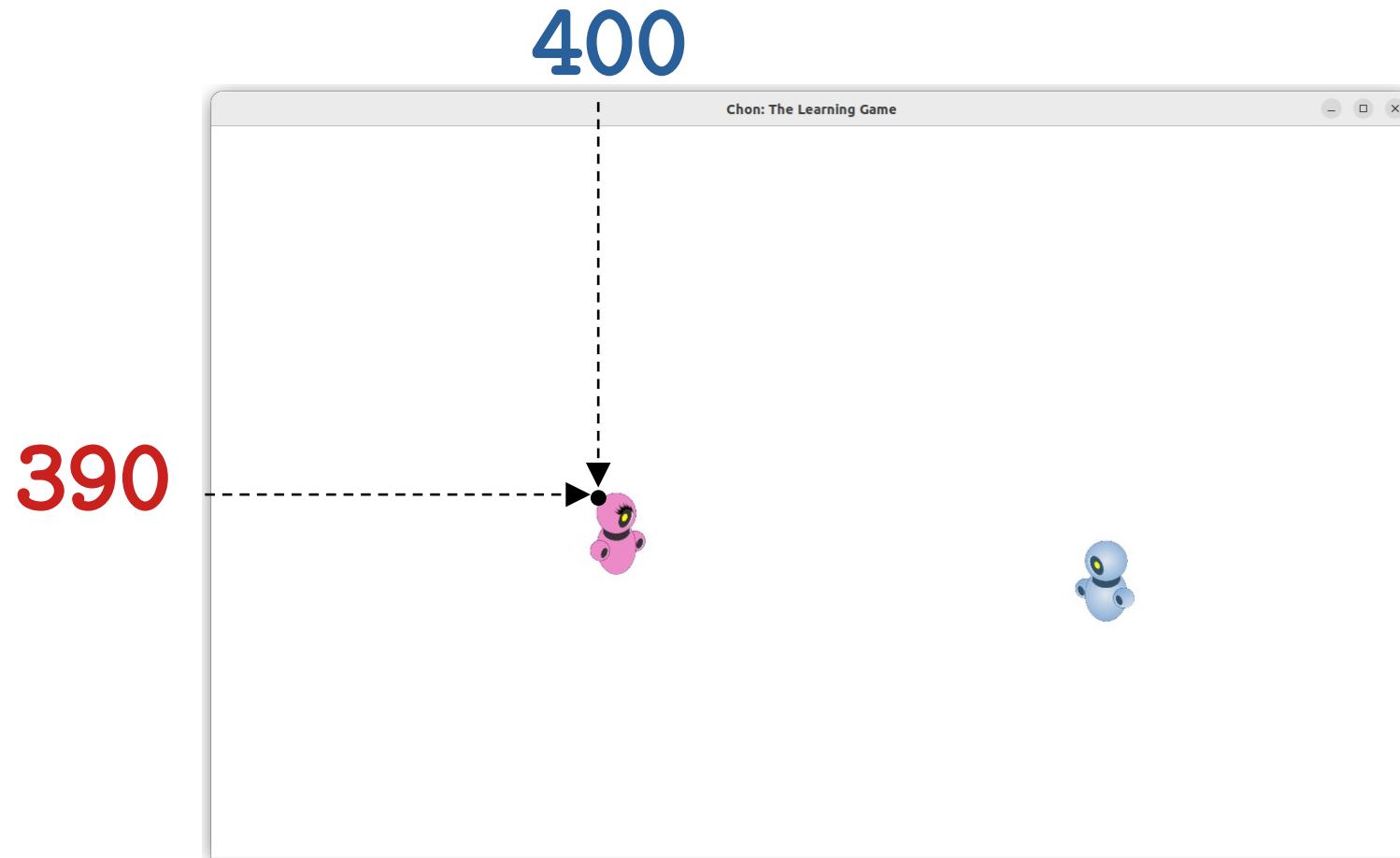
401

390

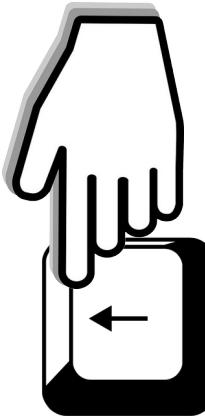
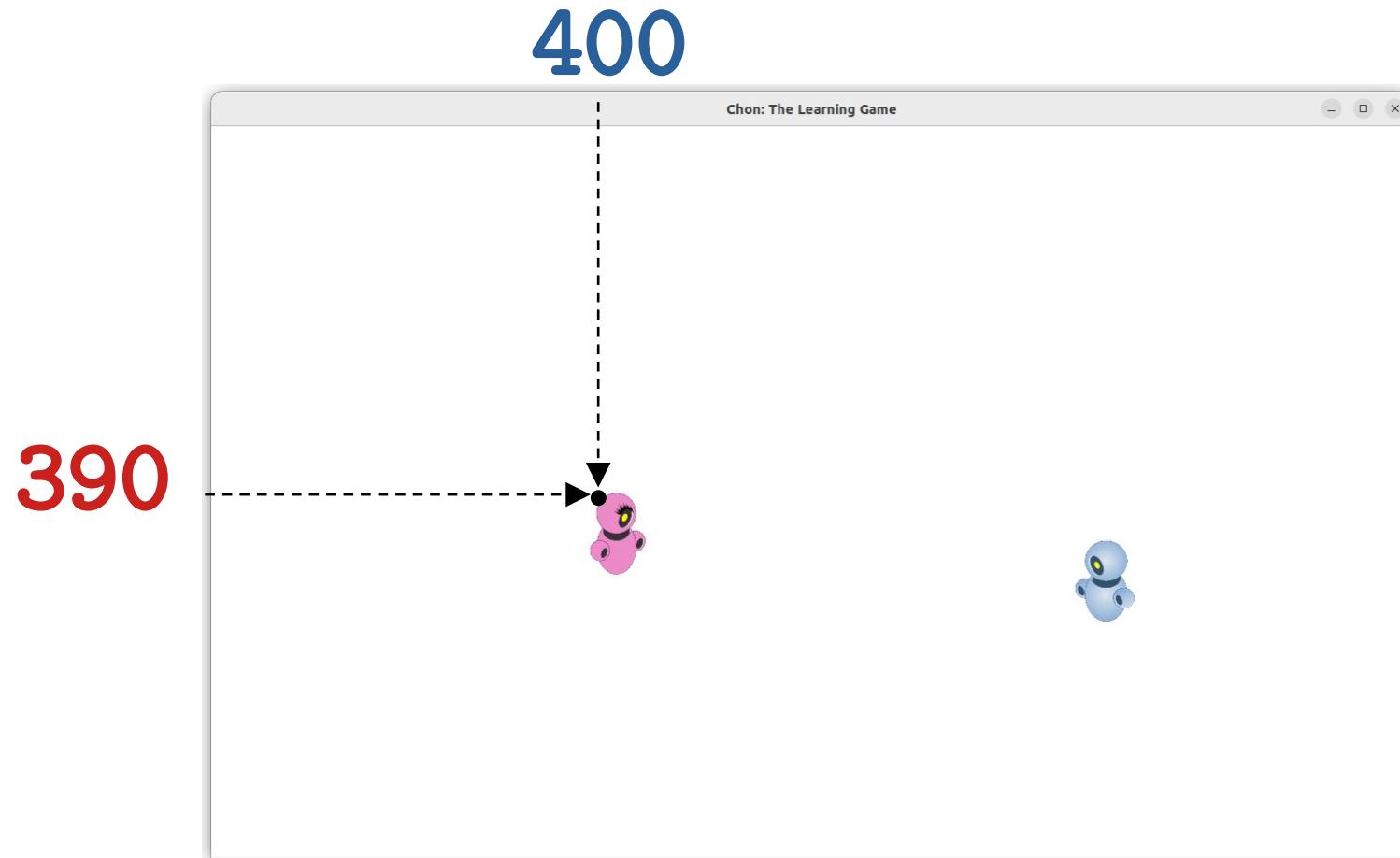


$x + \text{value}$

# The LEFT Logic

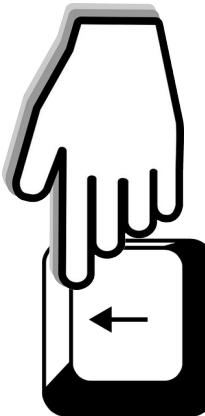
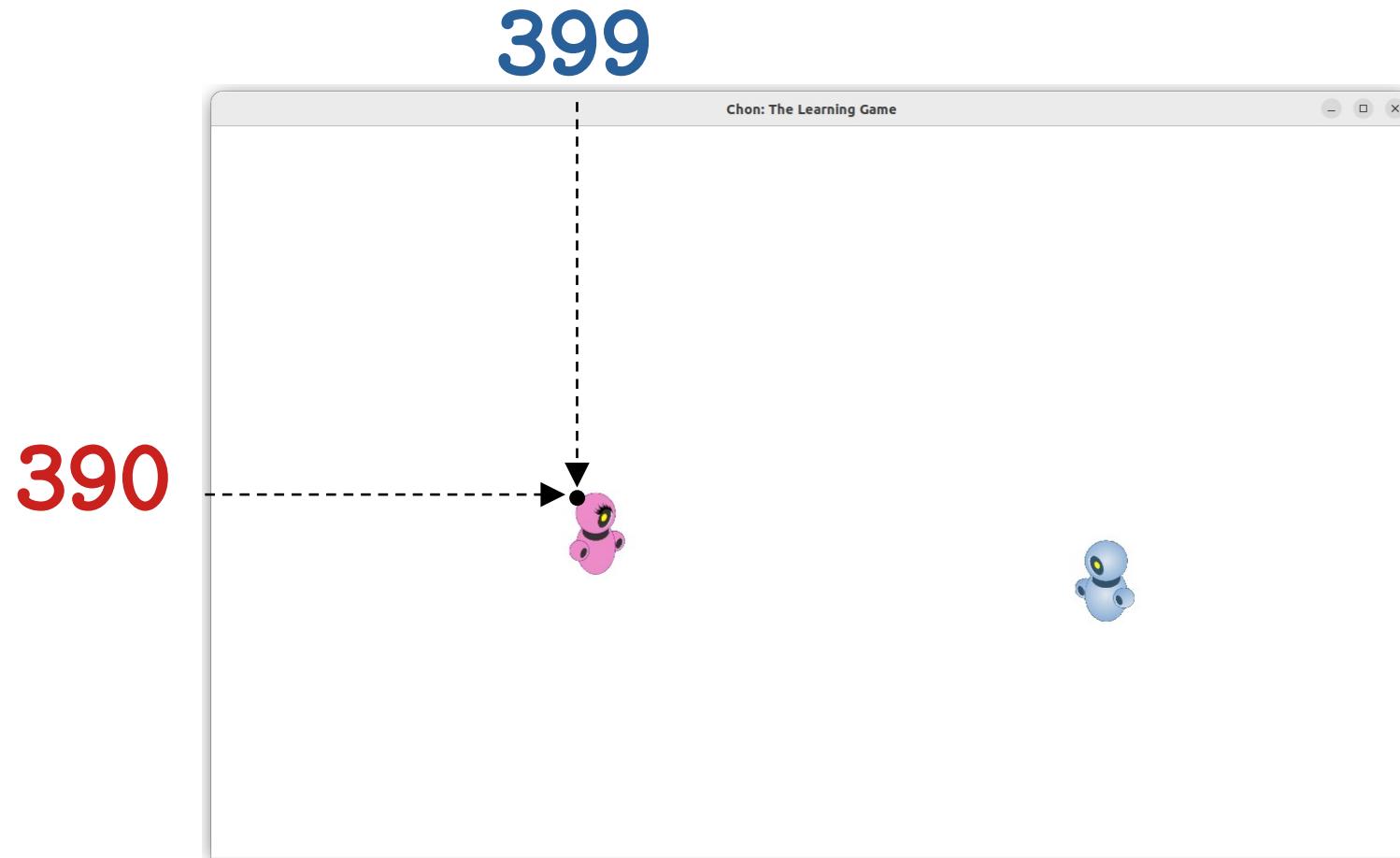


# The LEFT Logic



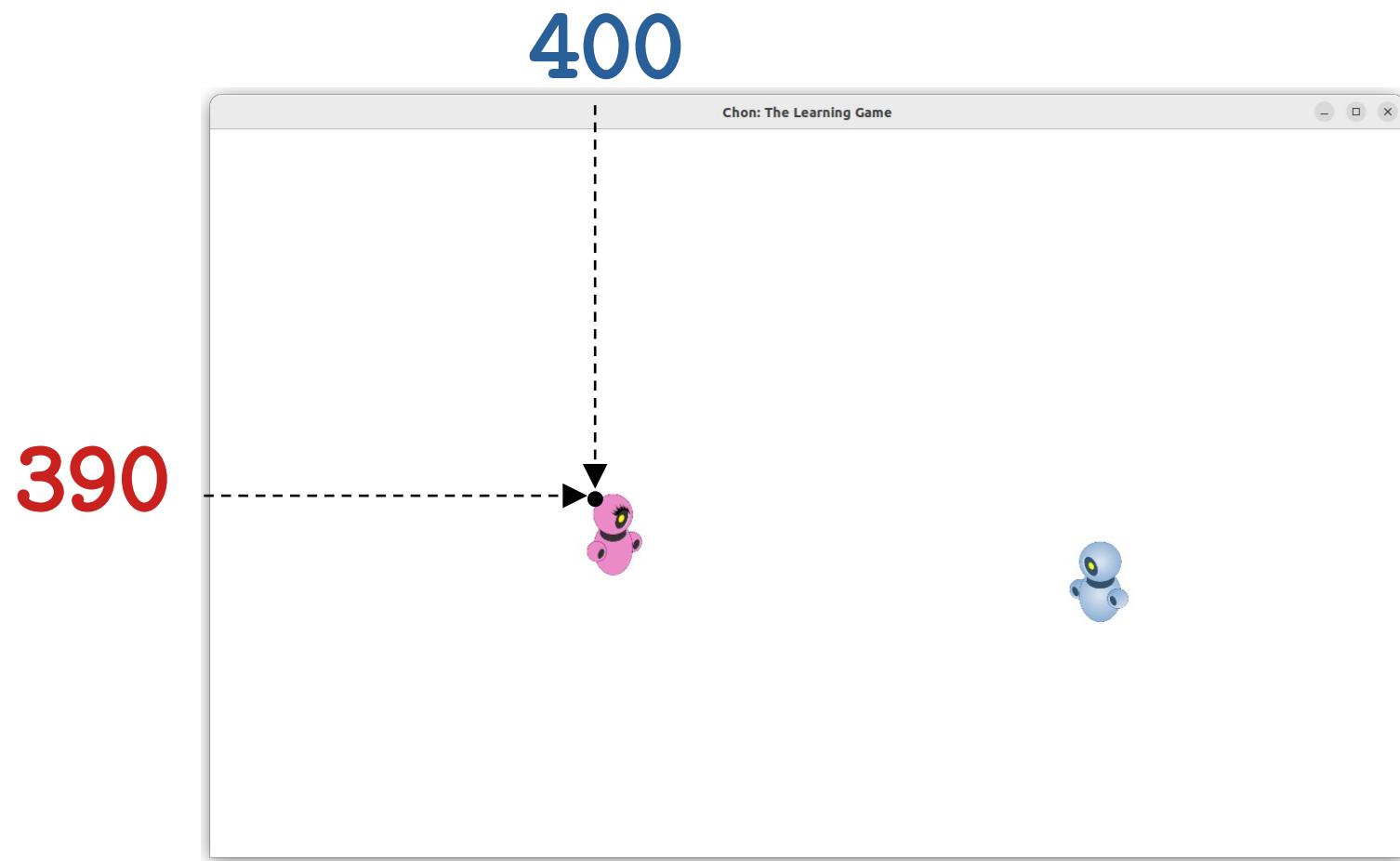
x - value

# The LEFT Logic

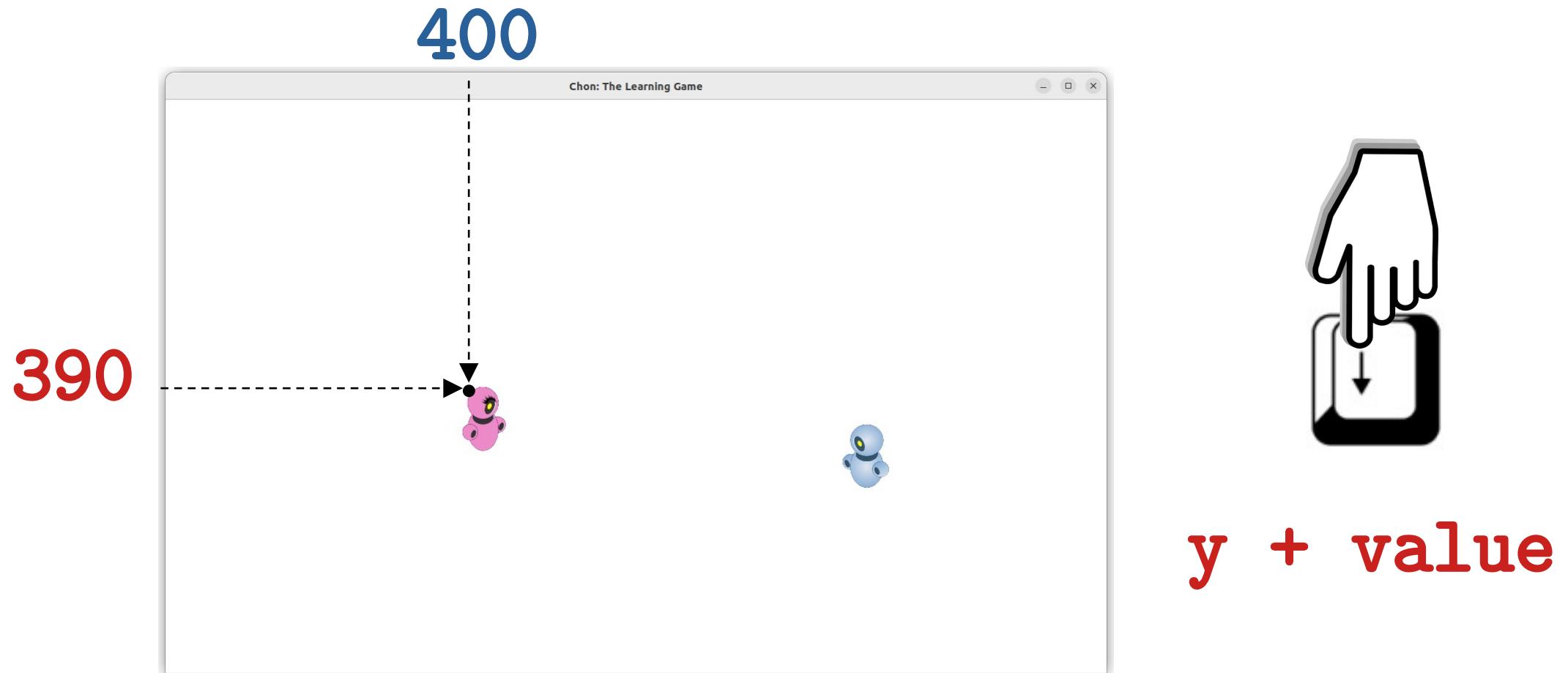


x - value

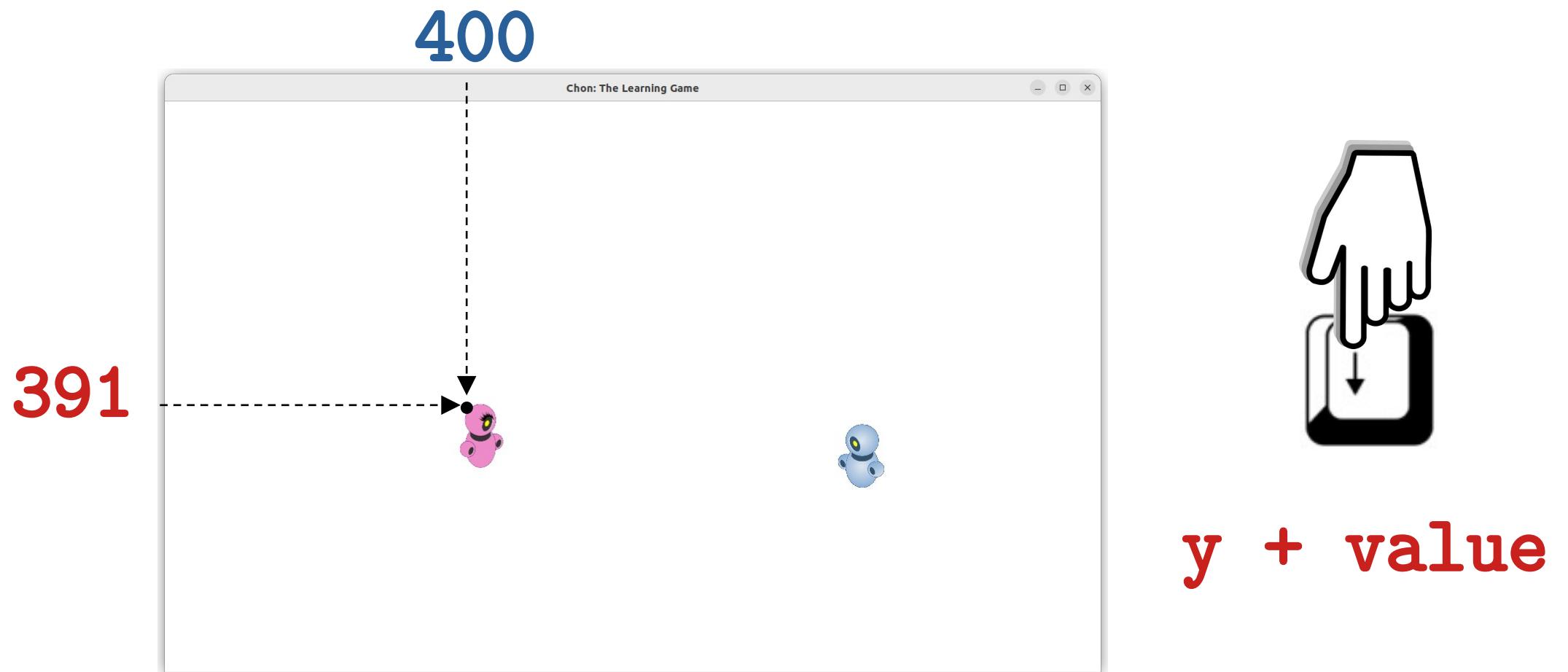
# The DOWN Logic



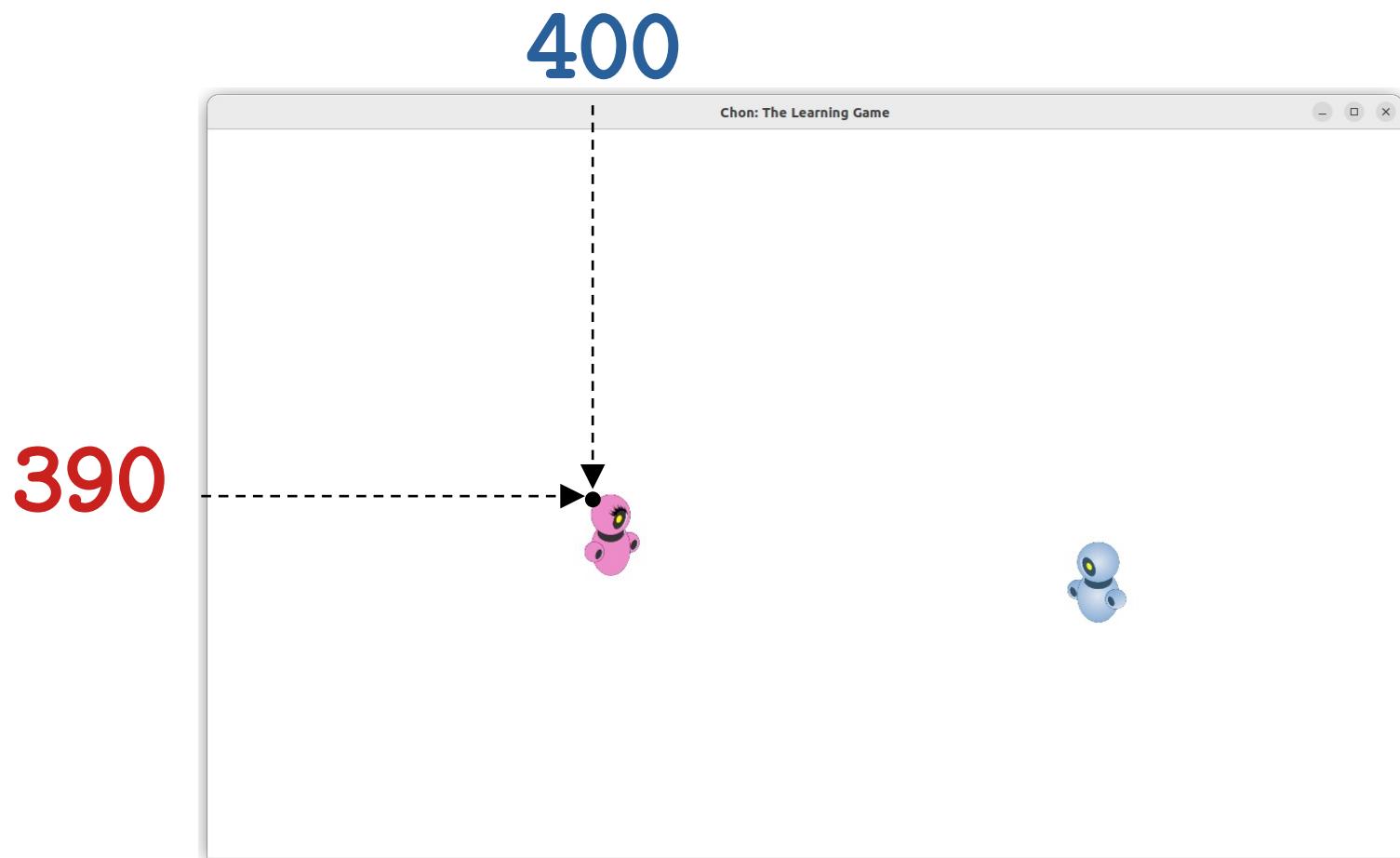
# The DOWN Logic



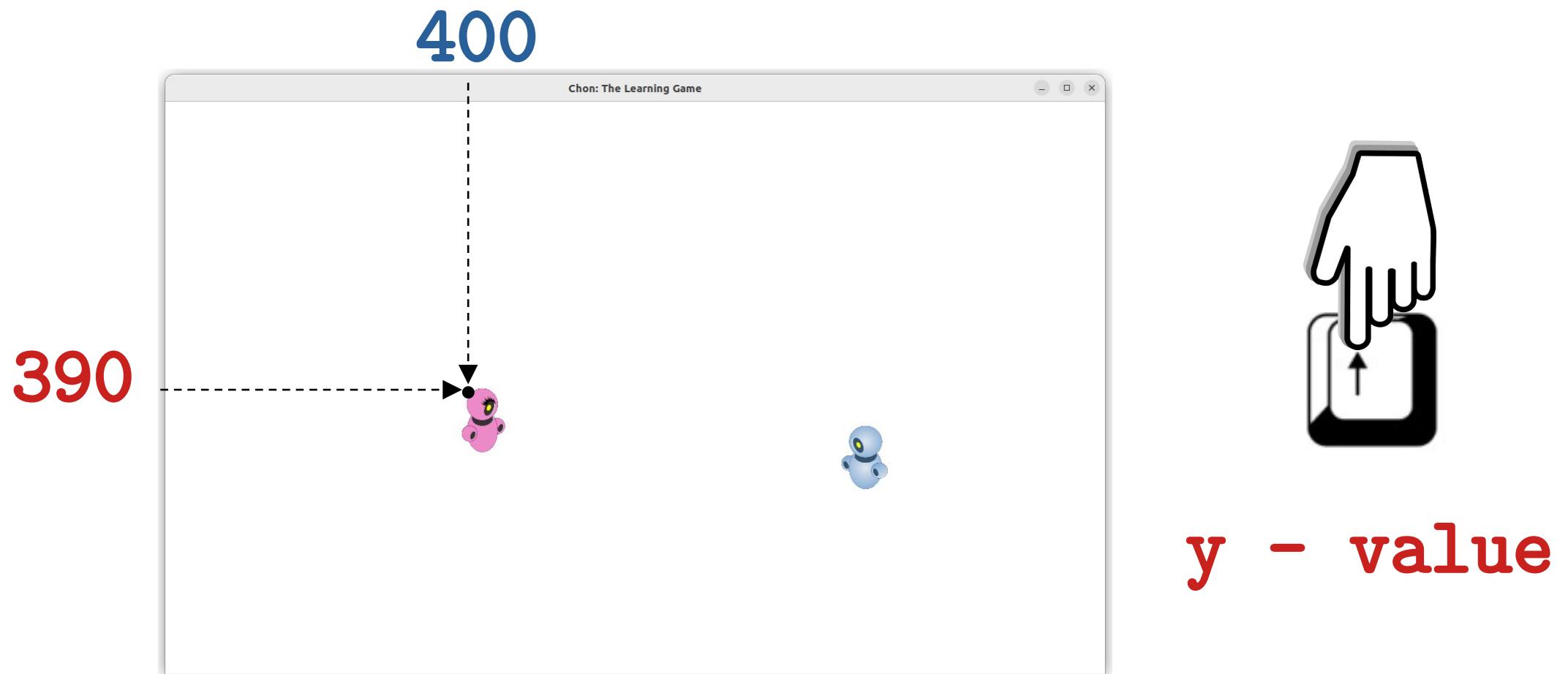
# The DOWN Logic



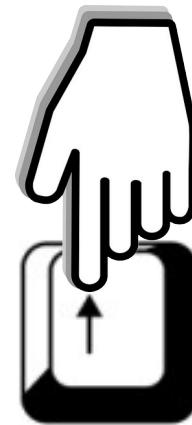
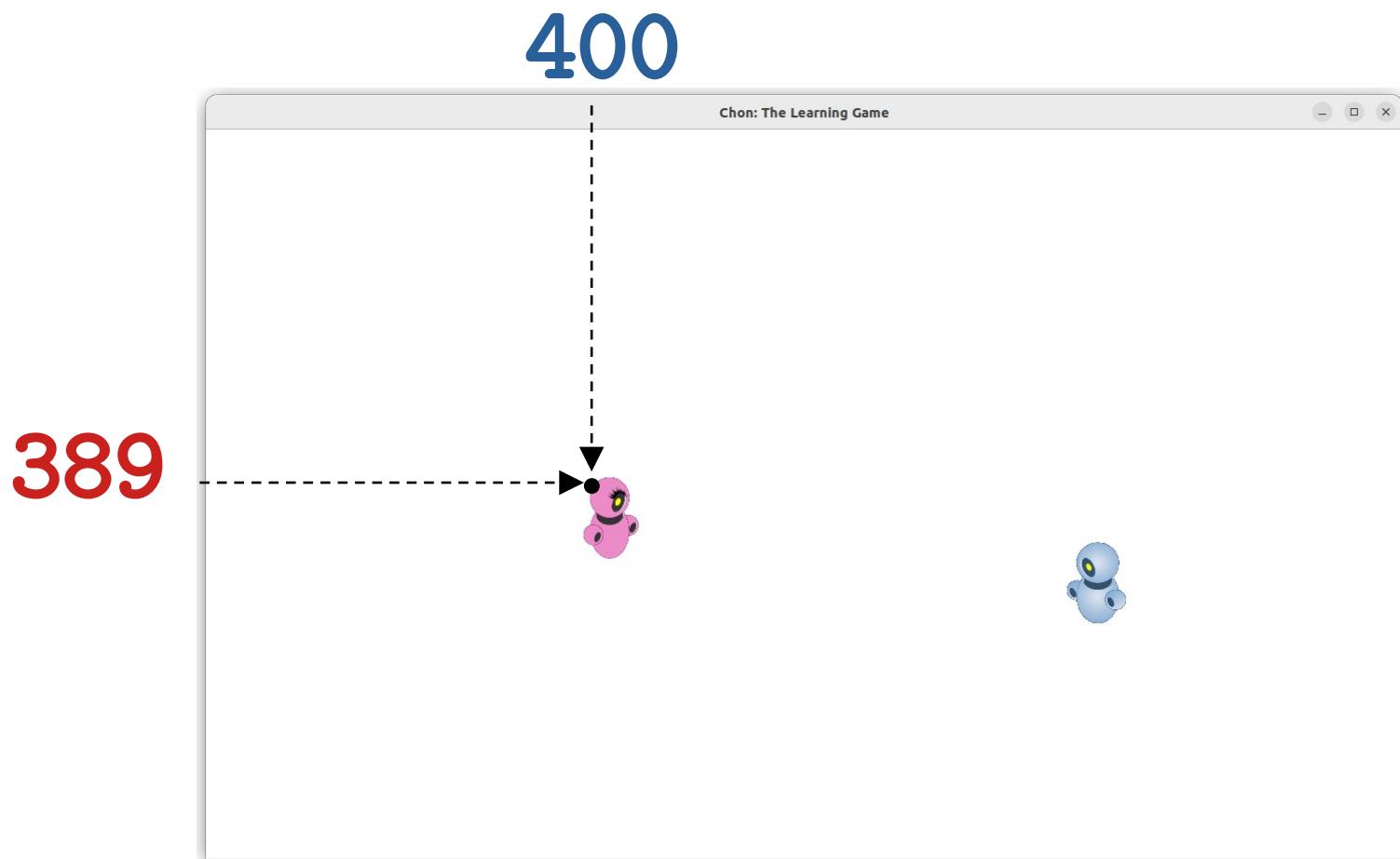
# The UP Logic



# The UP Logic



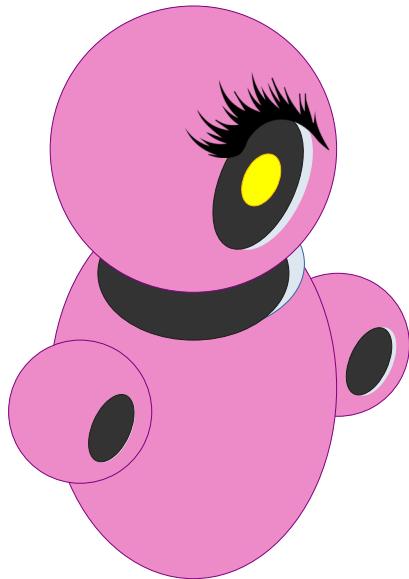
# The UP Logic



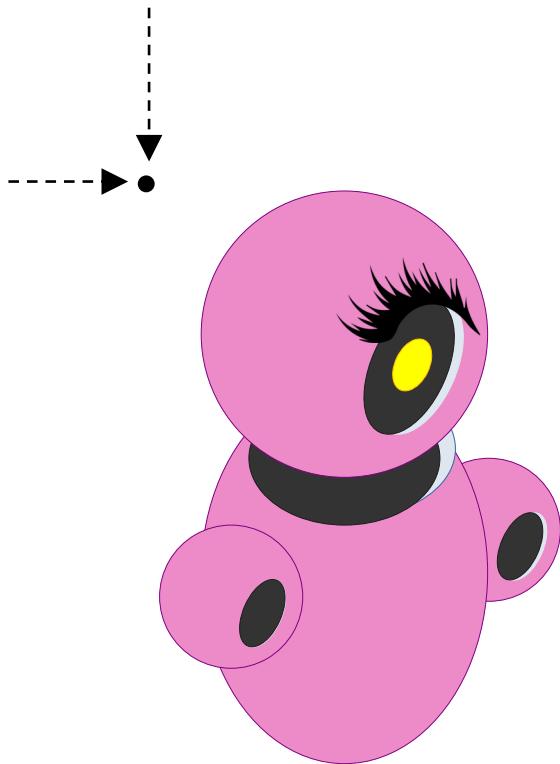
y - value

# Defining Boundaries

Every Image has:



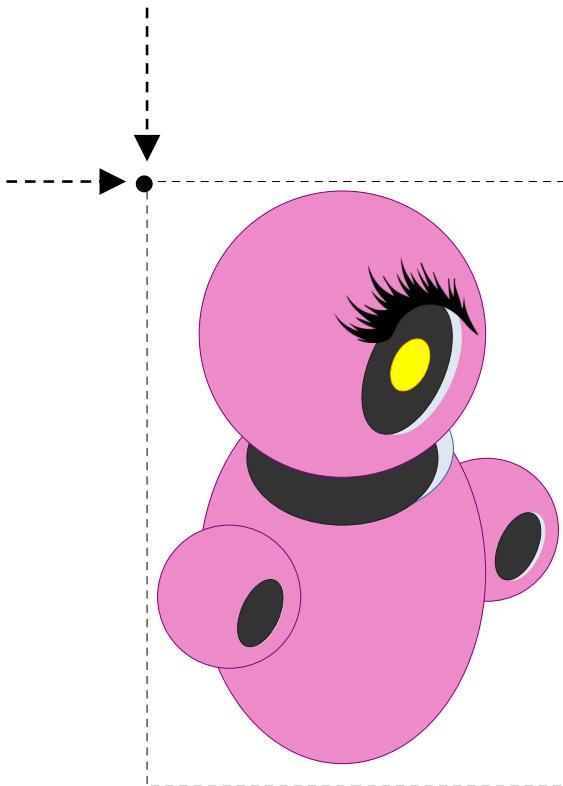
# Defining Boundaries



Every **Image** has:

- x and y points;

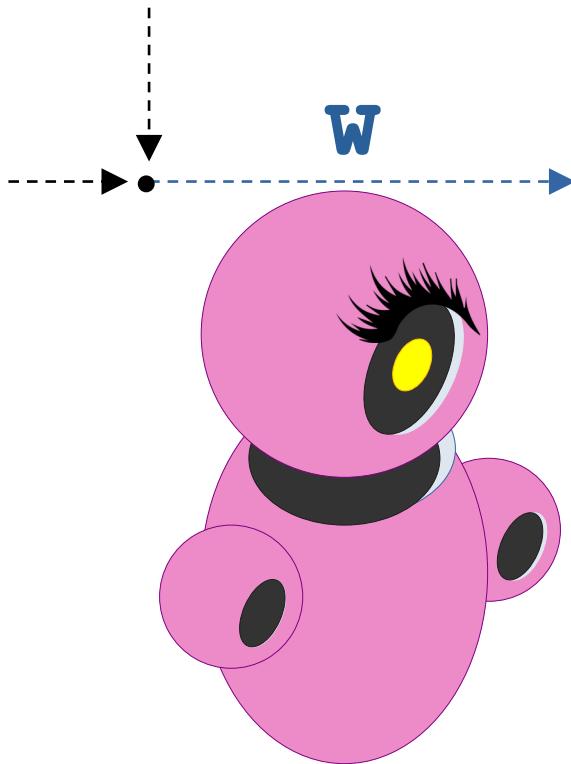
# Defining Boundaries



Every Image has:

- x and y points;

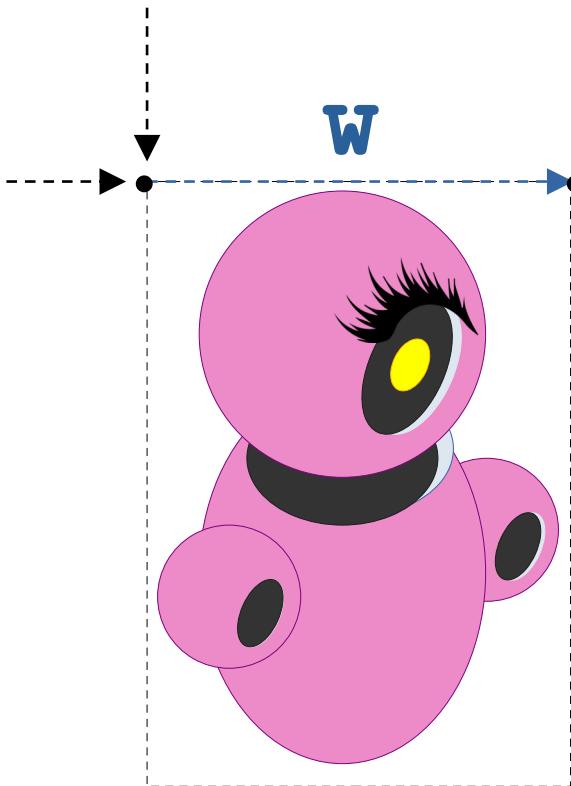
# Defining Boundaries



Every Image has:

- x and y points;
- Width;

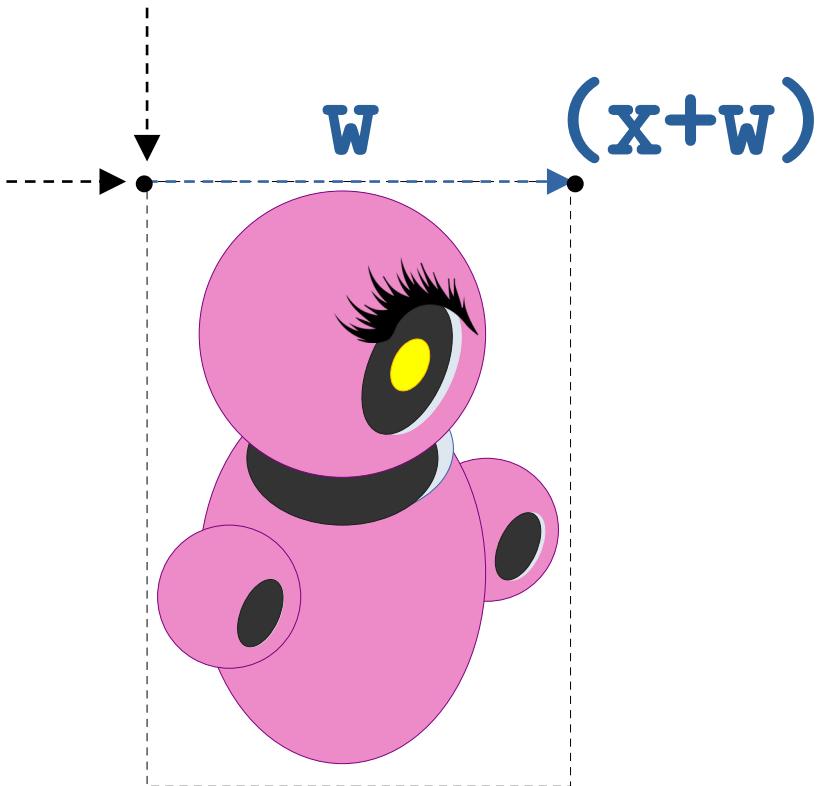
# Defining Boundaries



Every Image has:

- x and y points;
- Width;

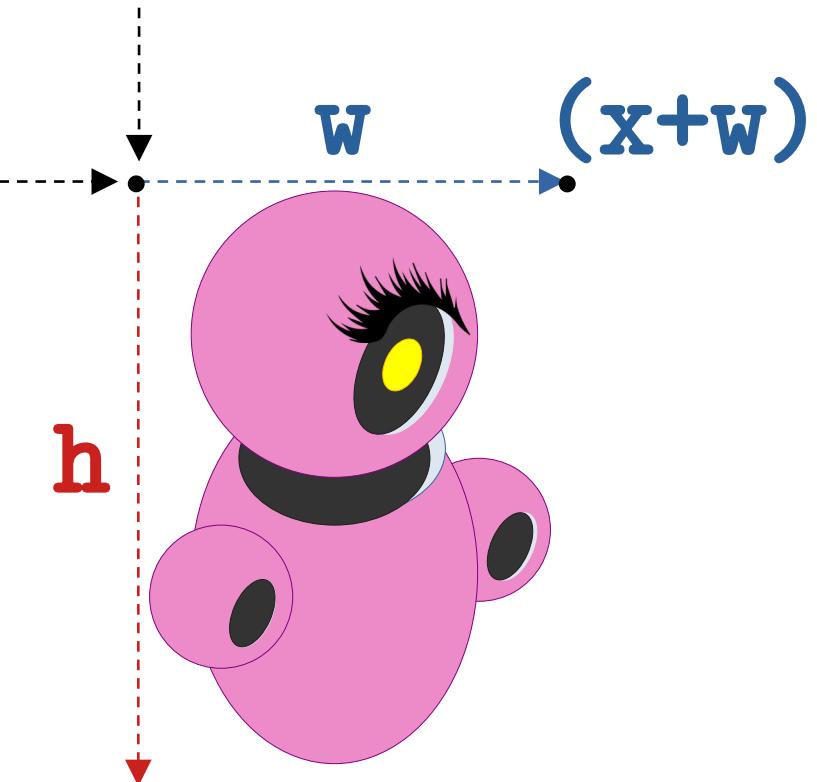
# Defining Boundaries



Every Image has:

- x and y points;
- Width;

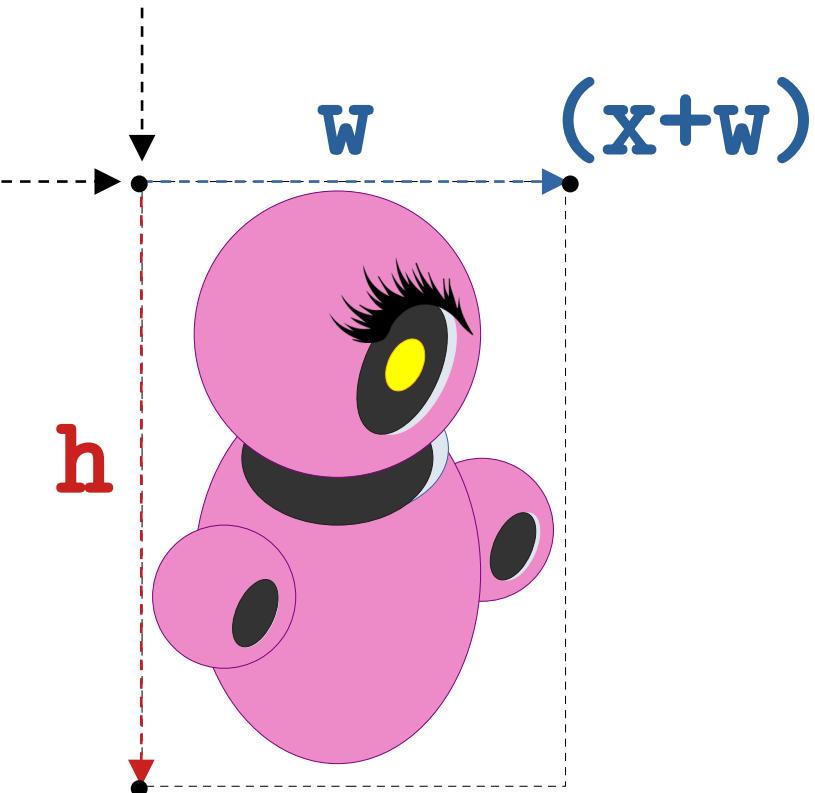
# Defining Boundaries



Every Image has:

- x and y points;
  - Width;
  - Height.

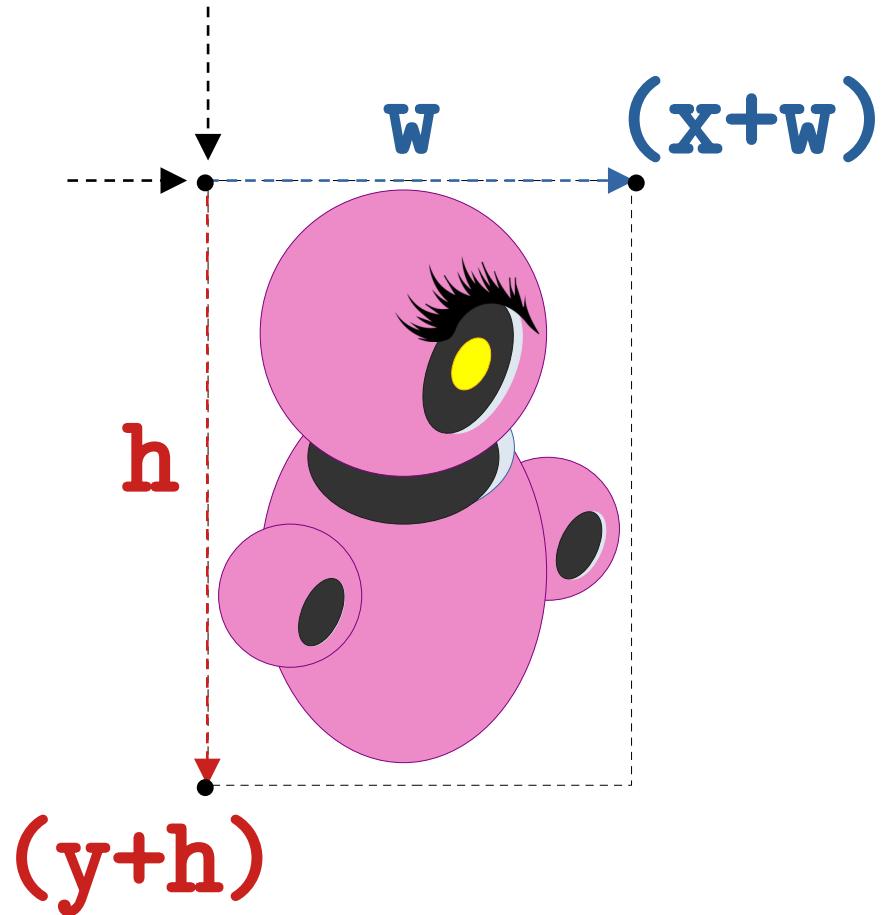
# Defining Boundaries



Every Image has:

- x and y points;
  - Width;
  - Height.

# Defining Boundaries



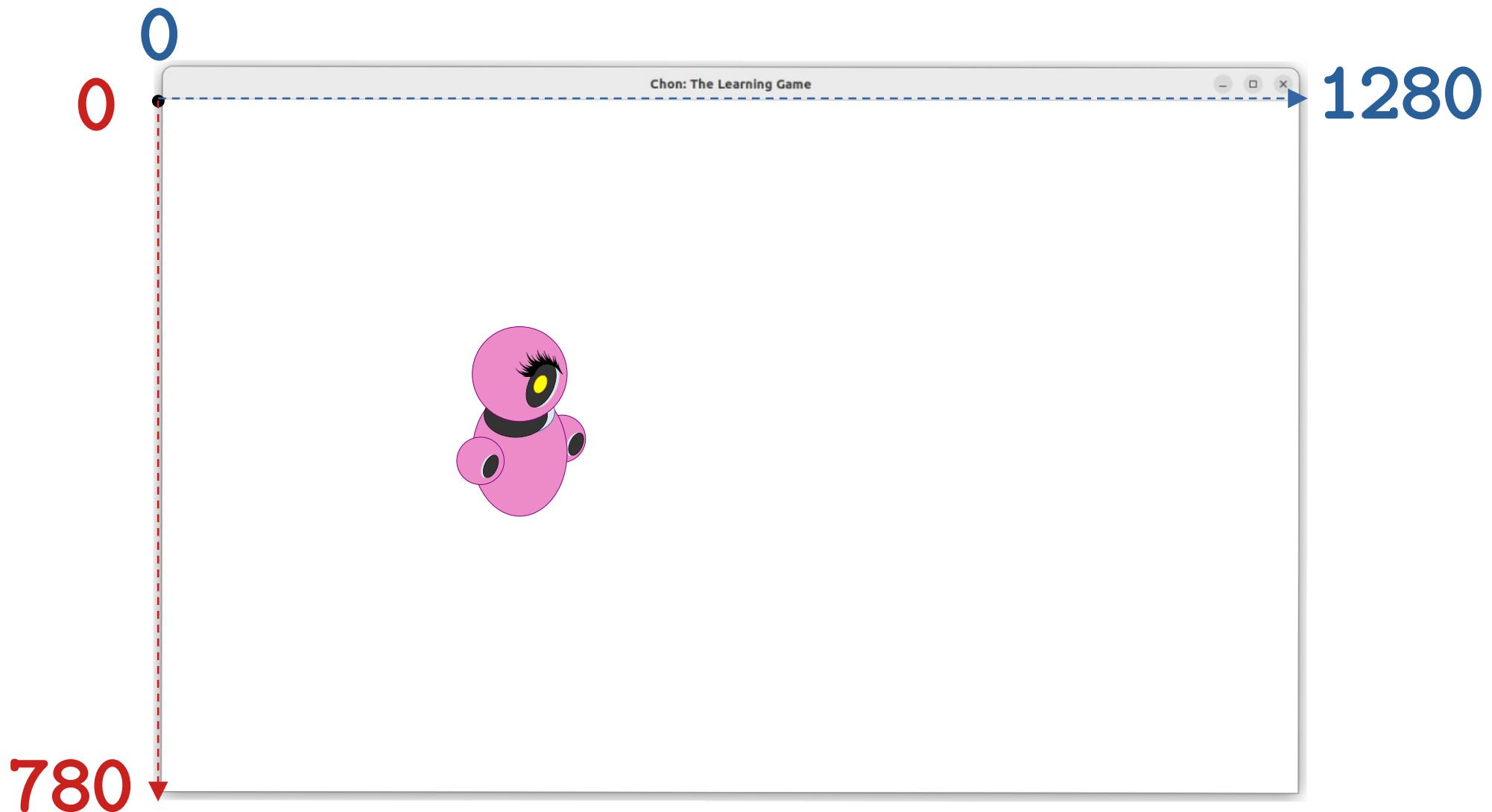
Every Image has:

- x and y points;
  - Width;
  - Height.

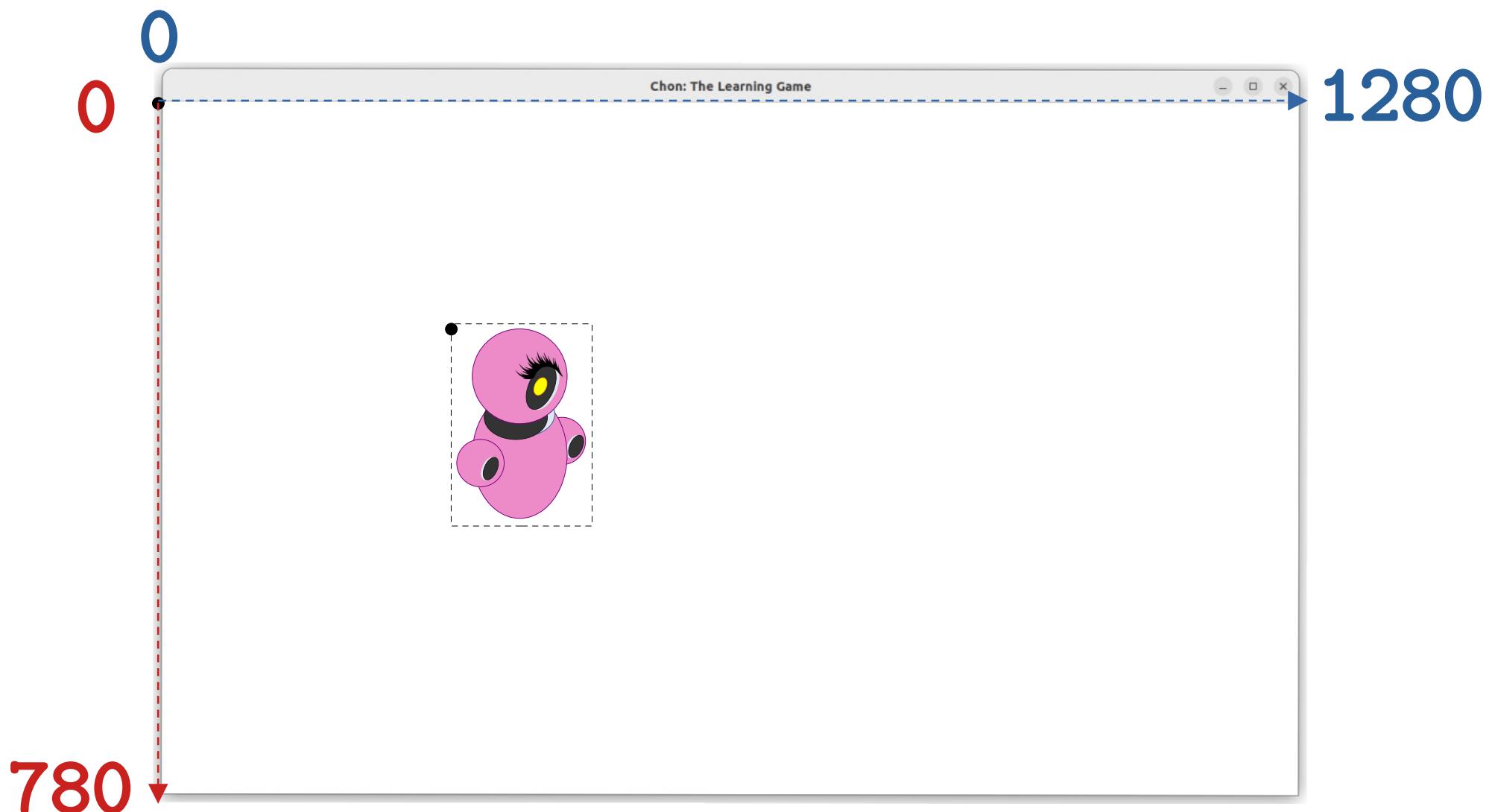
# Defining Boundaries



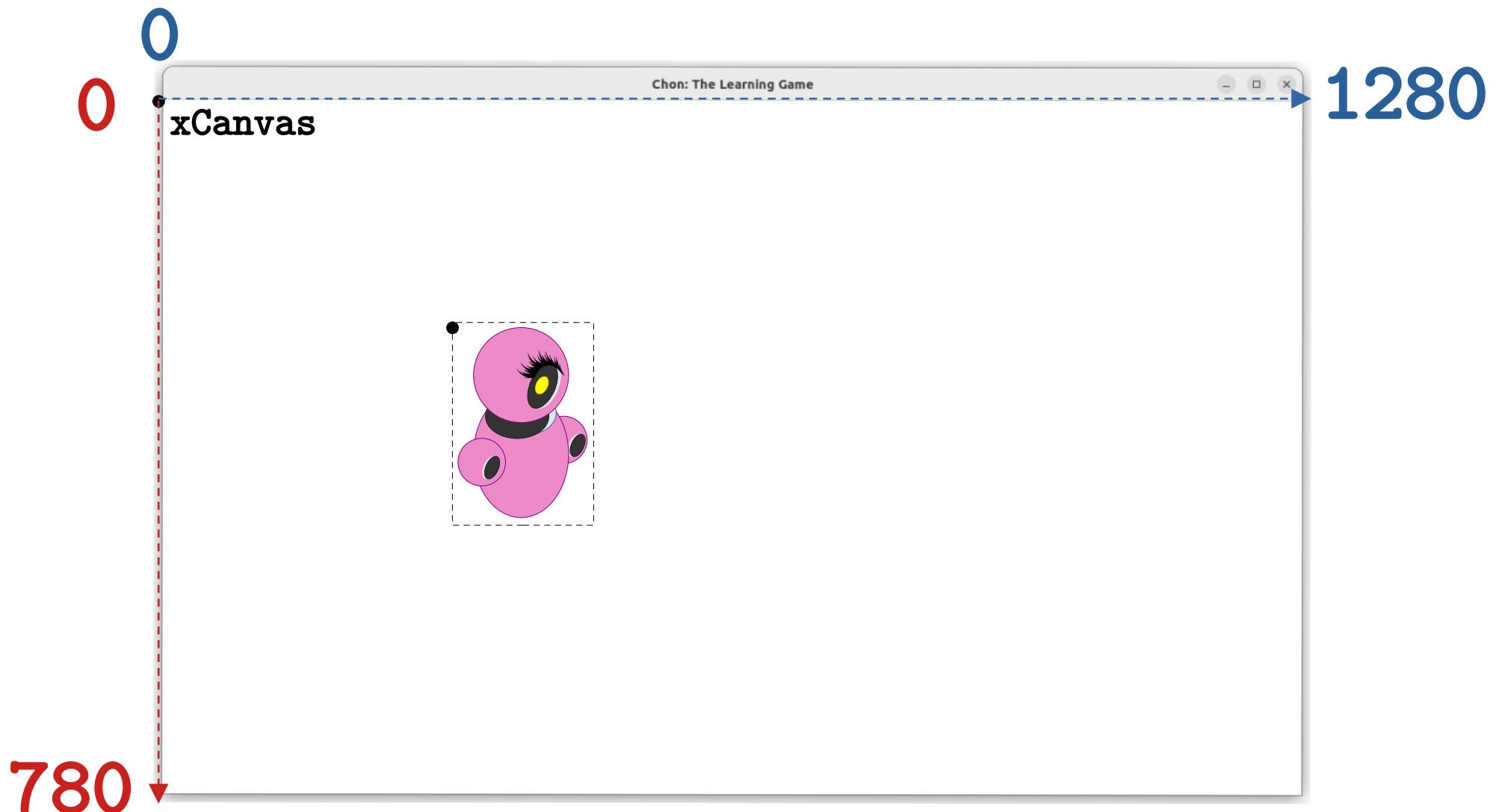
# Defining Boundaries



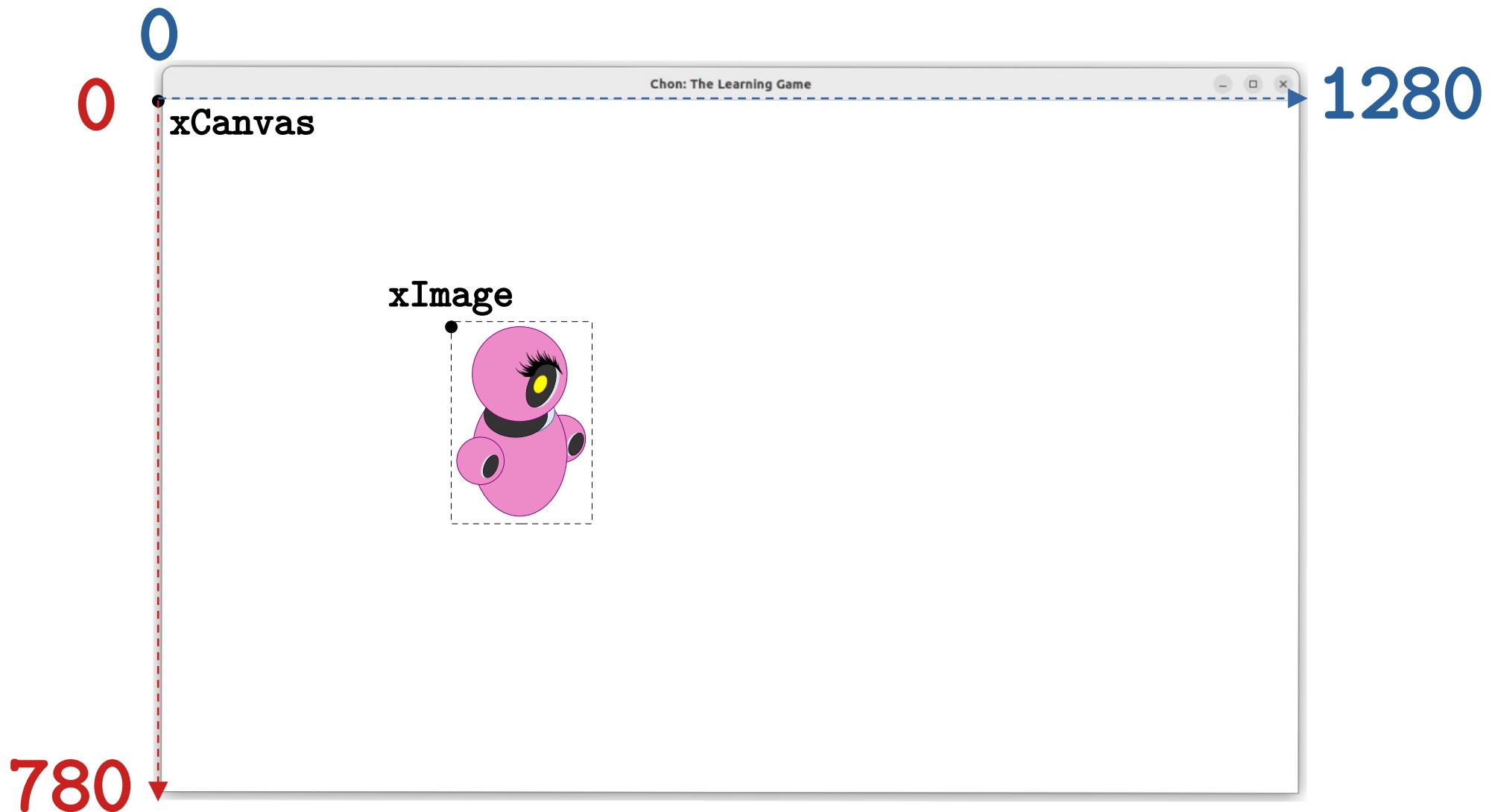
# Defining Boundaries at the LEFT



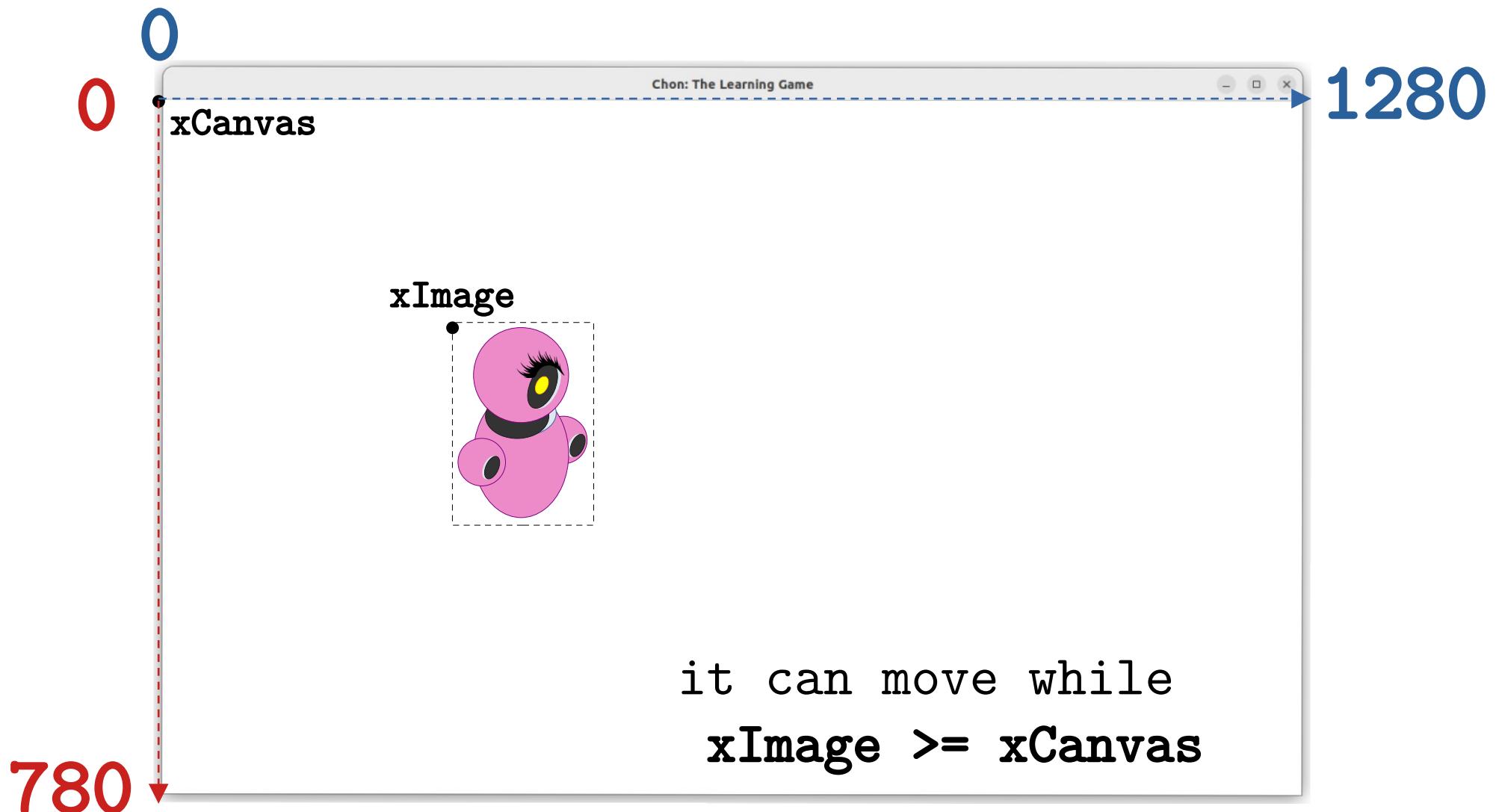
# Defining Boundaries at the LEFT



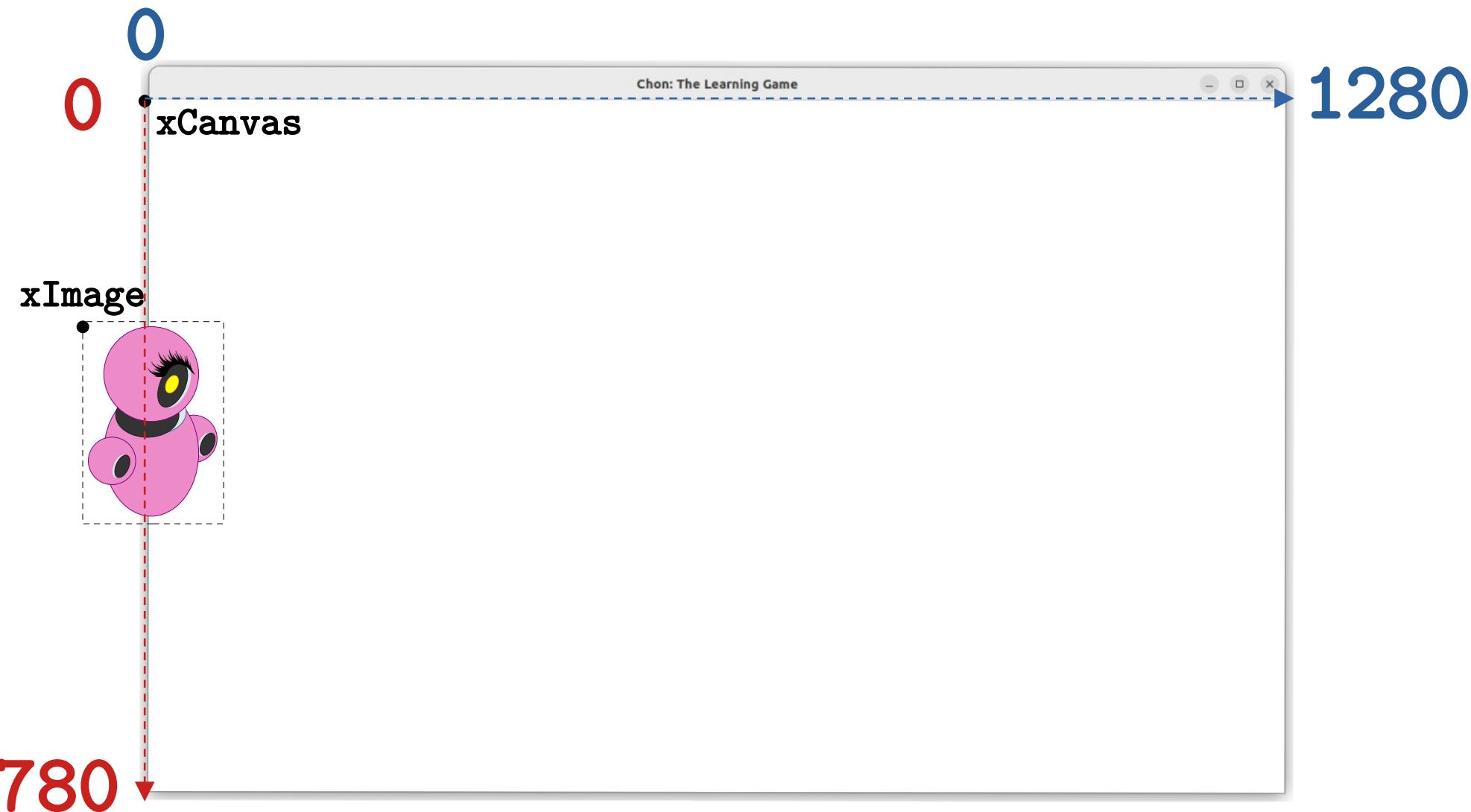
# Defining Boundaries at the LEFT



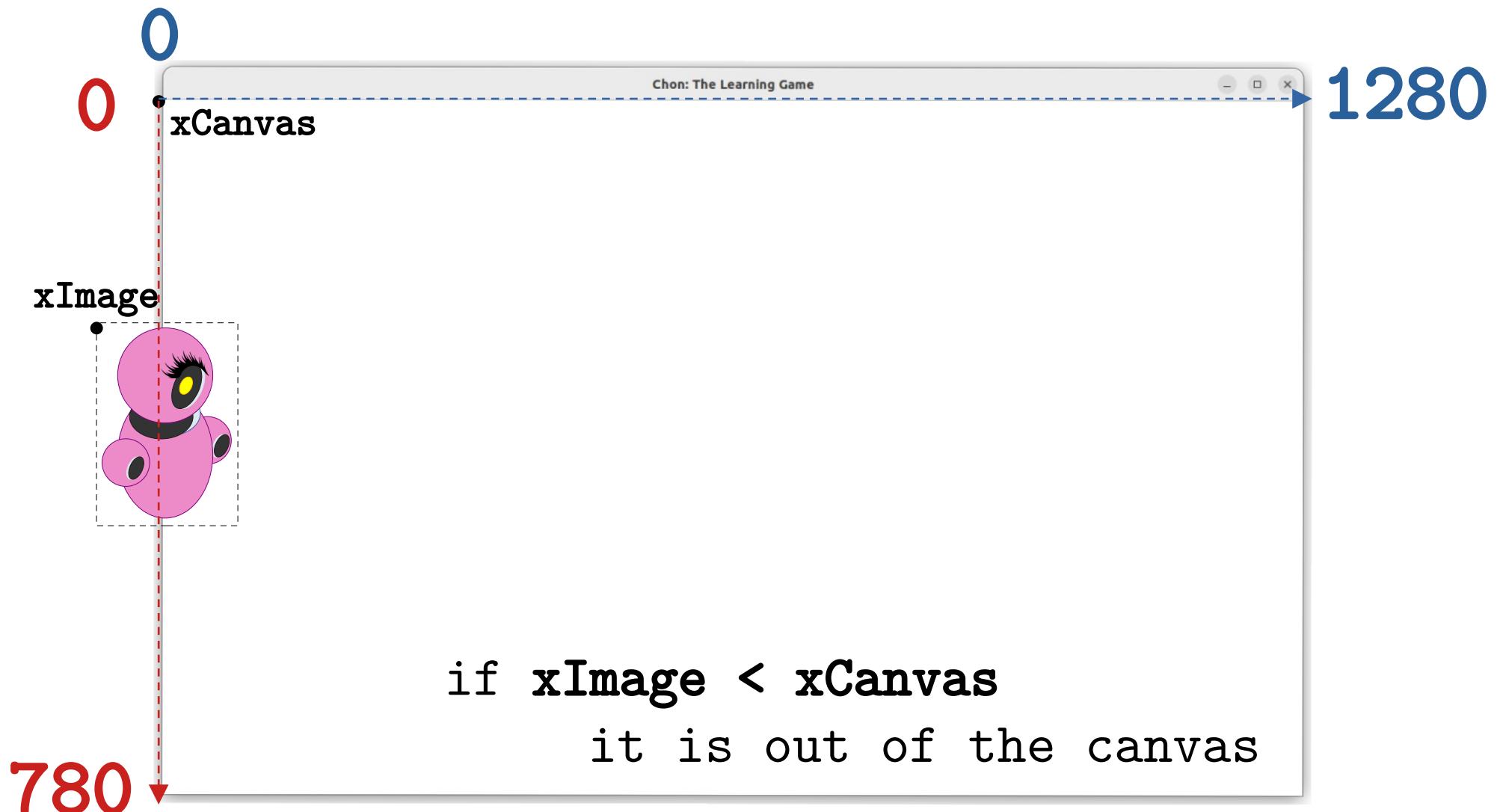
# Defining Boundaries at the LEFT



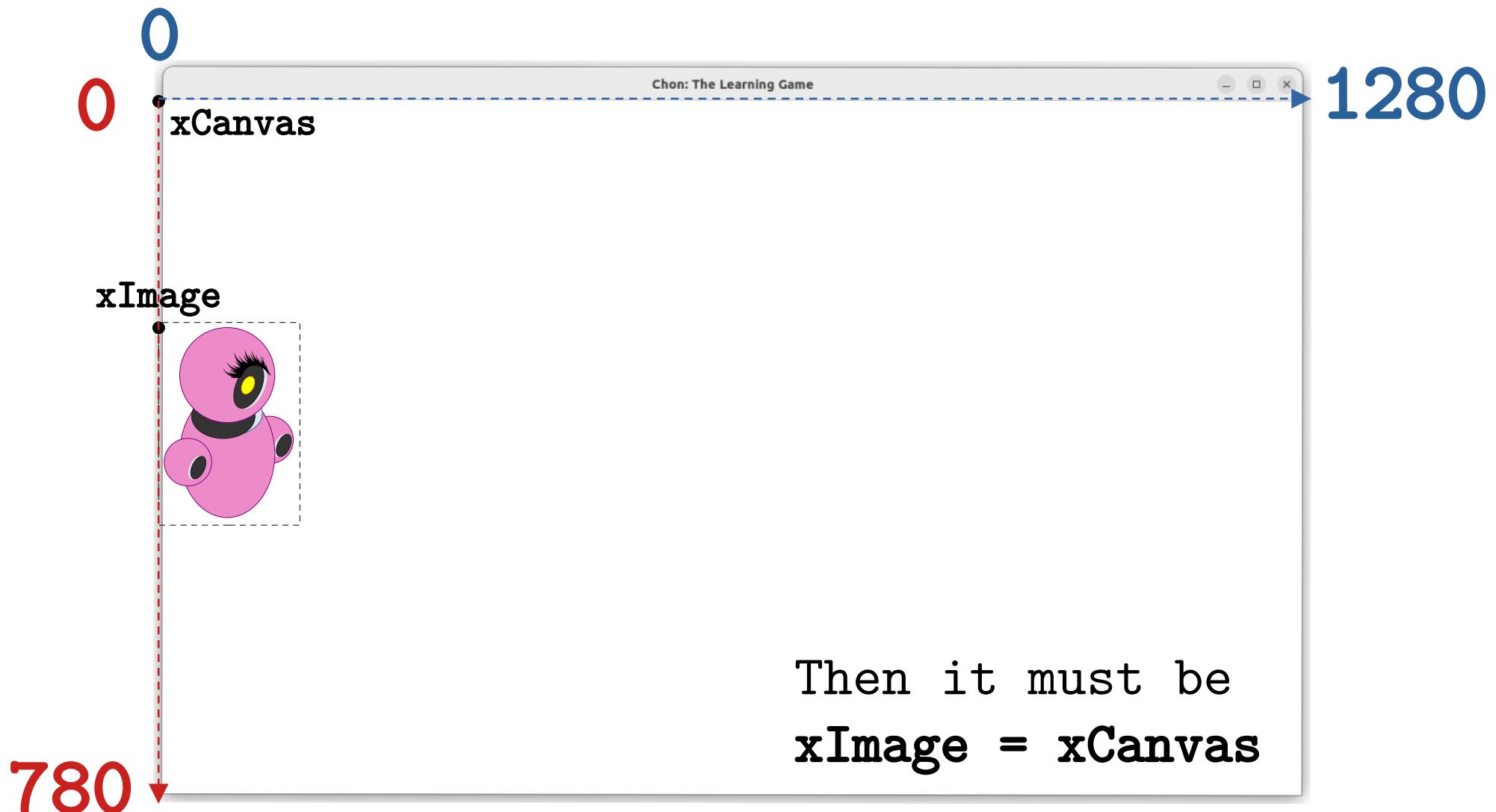
# Defining Boundaries at the LEFT



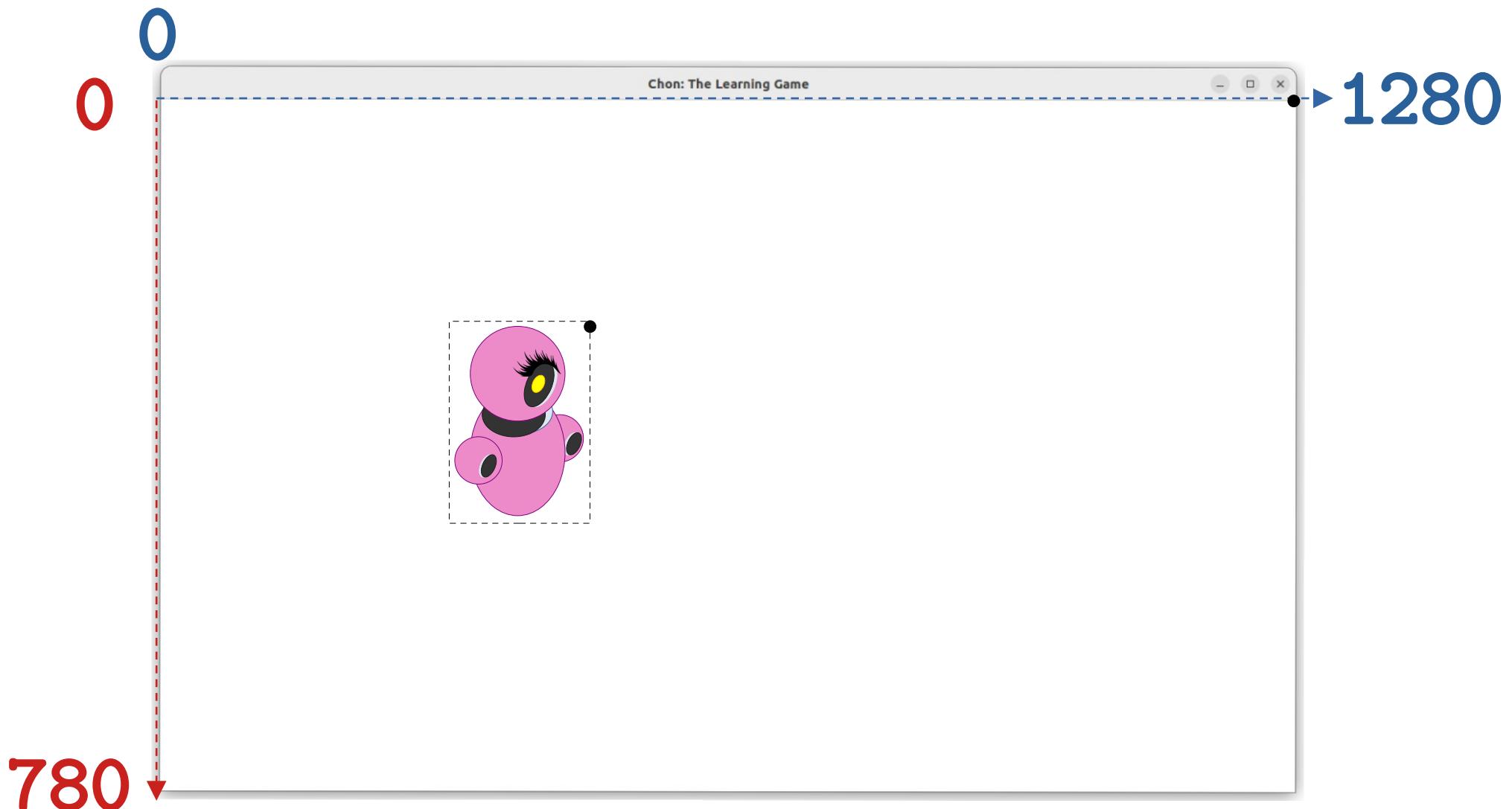
# Defining Boundaries at the LEFT



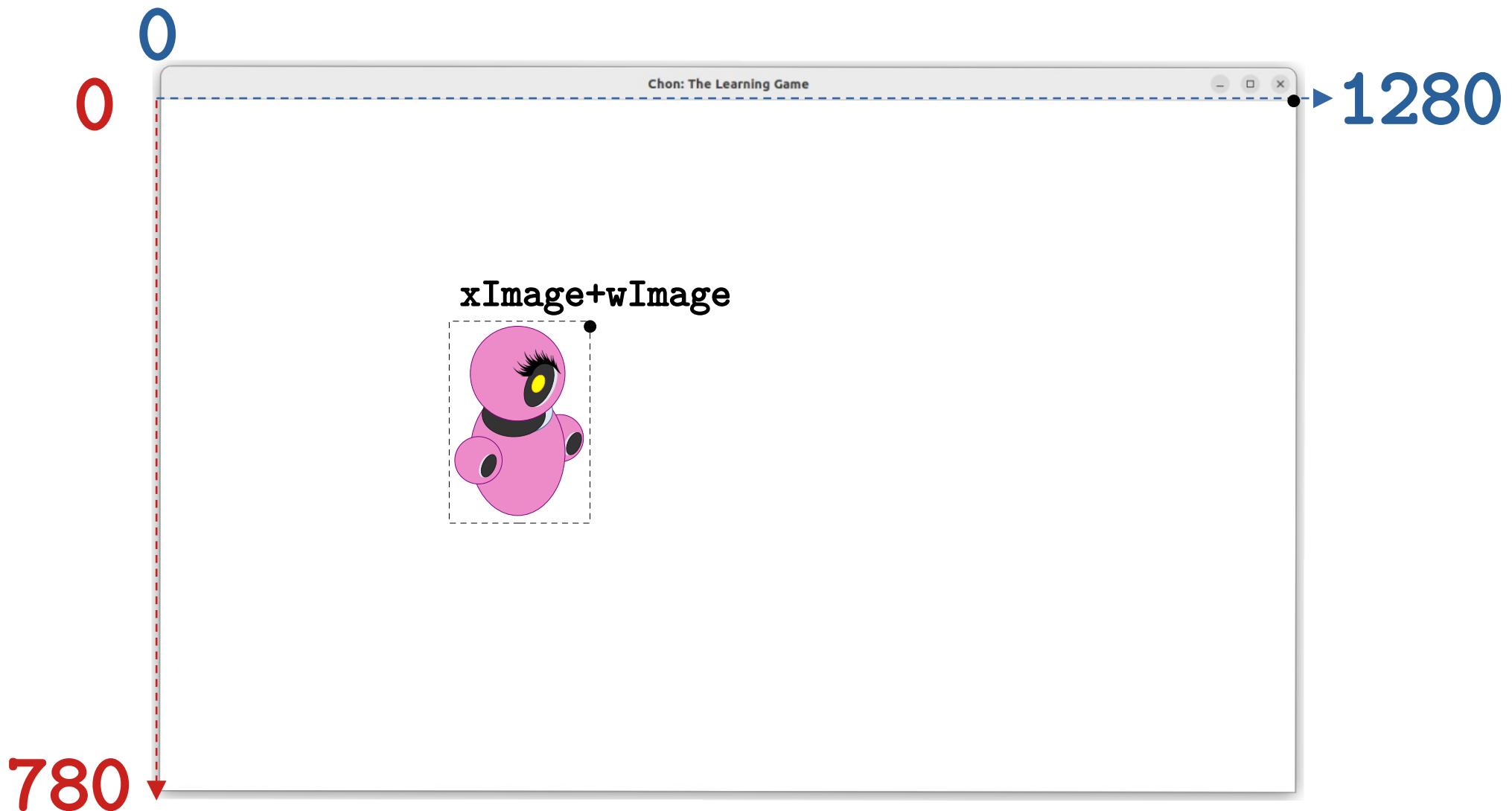
# Defining Boundaries at the LEFT



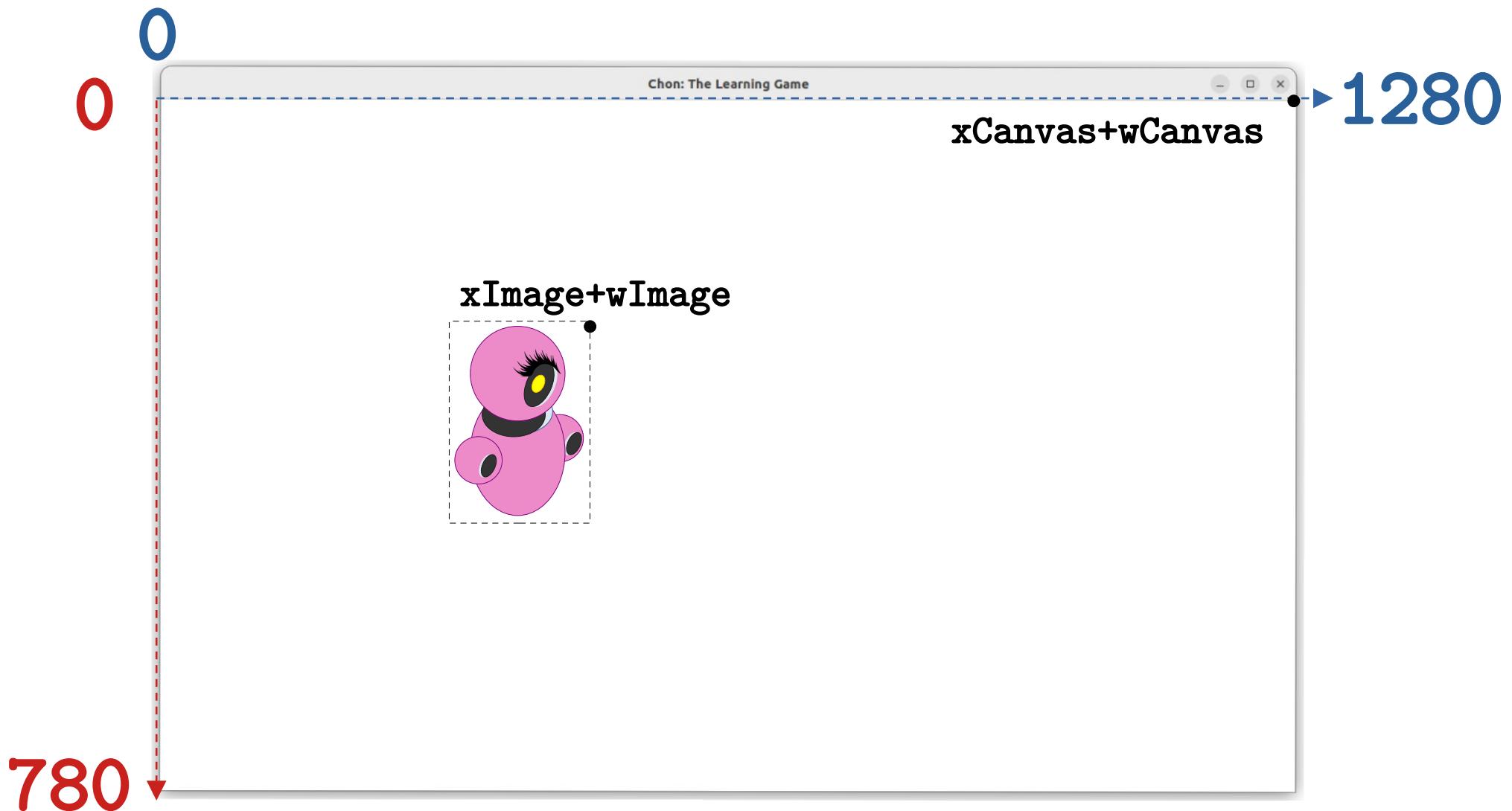
# Defining Boundaries at the RIGHT



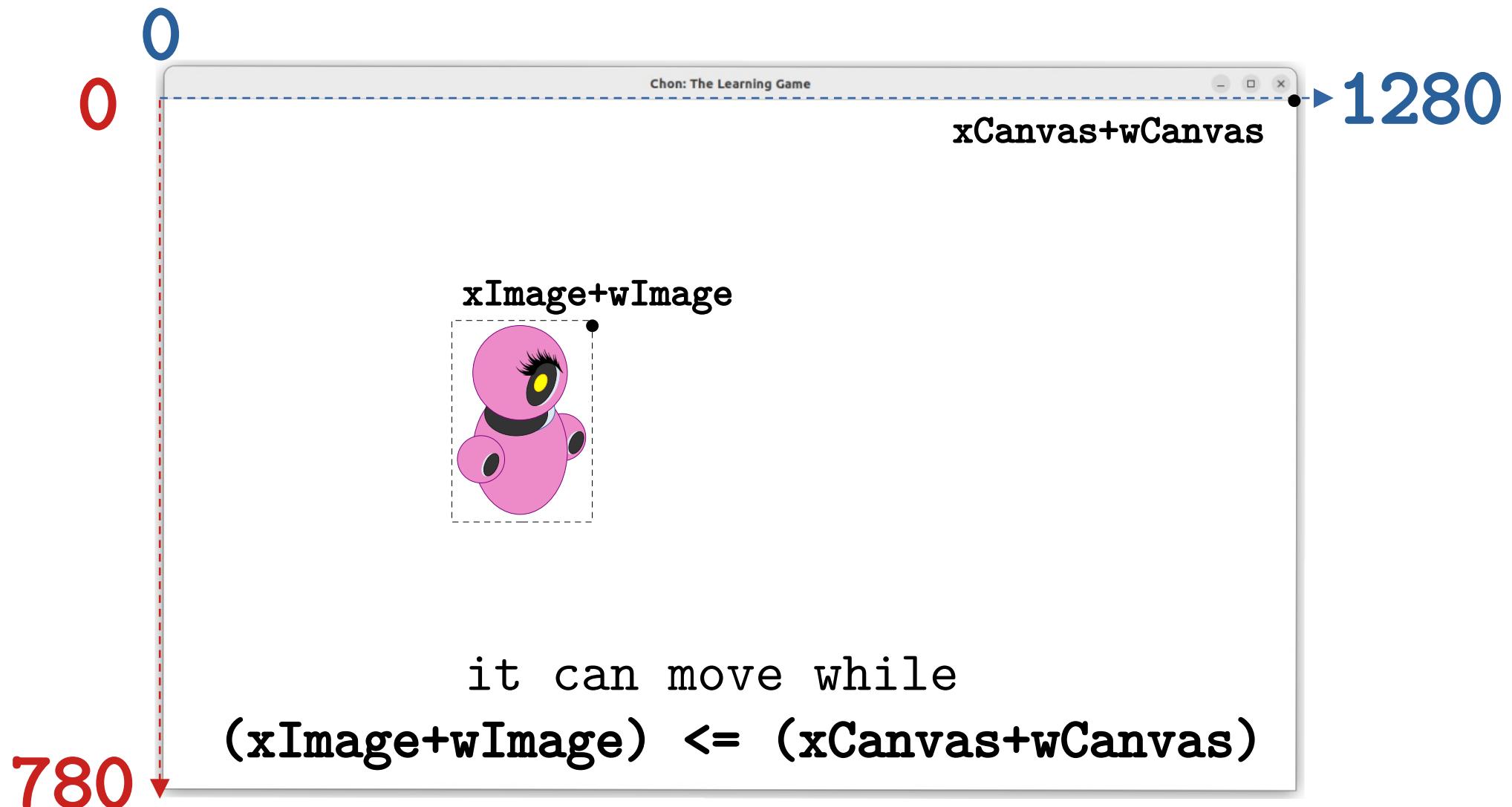
# Defining Boundaries at the RIGHT



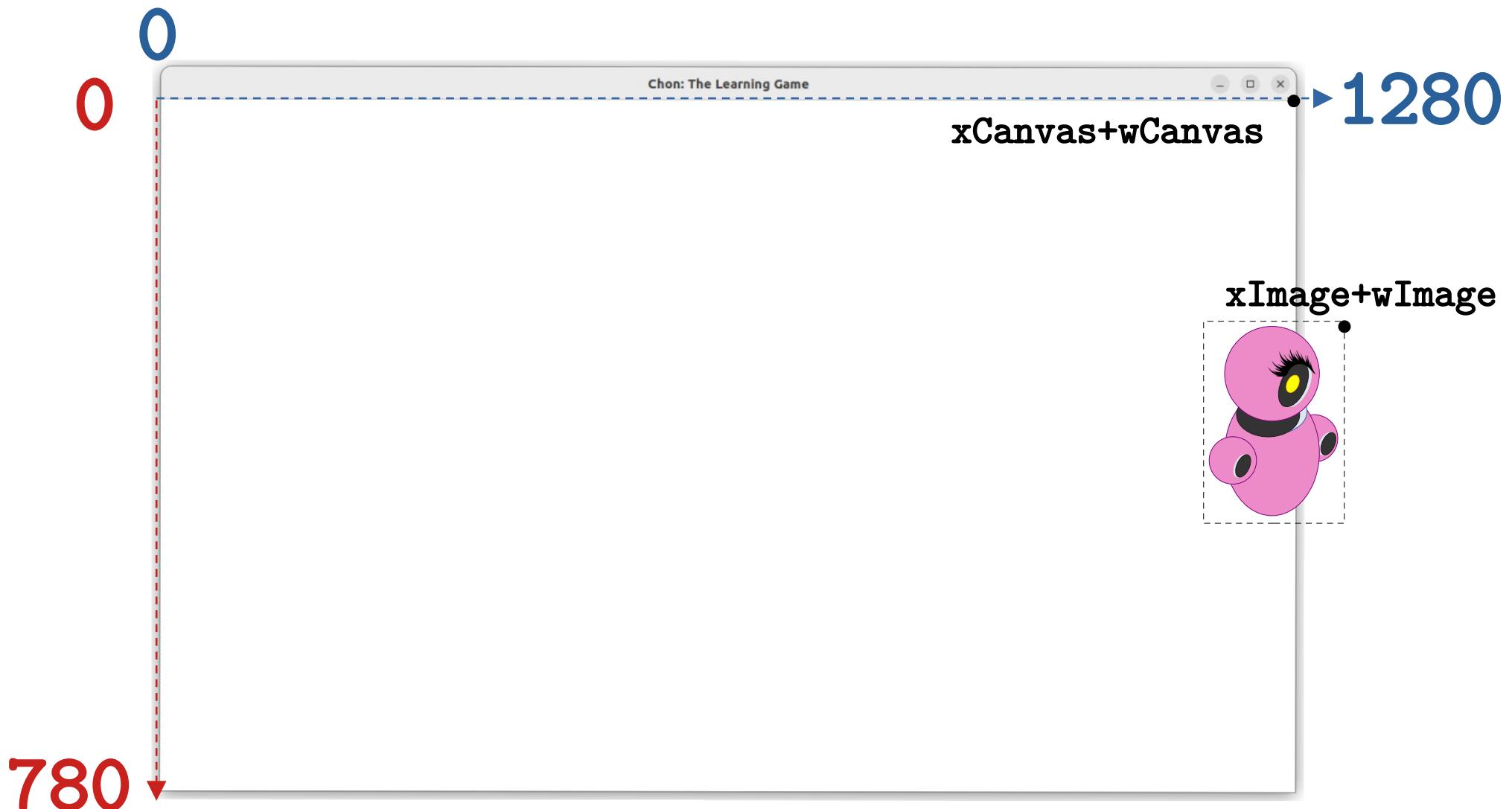
# Defining Boundaries at the RIGHT



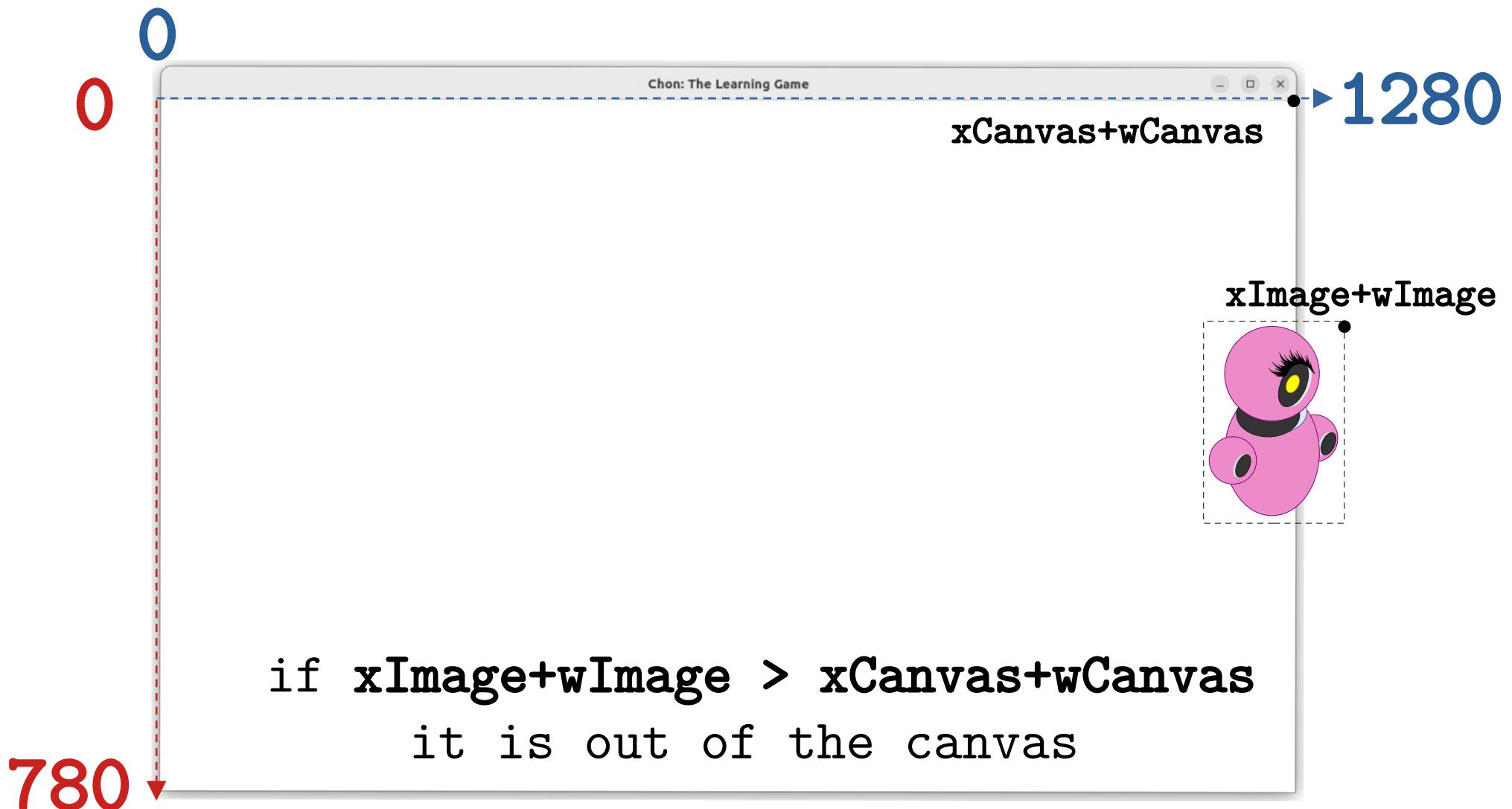
# Defining Boundaries at the RIGHT



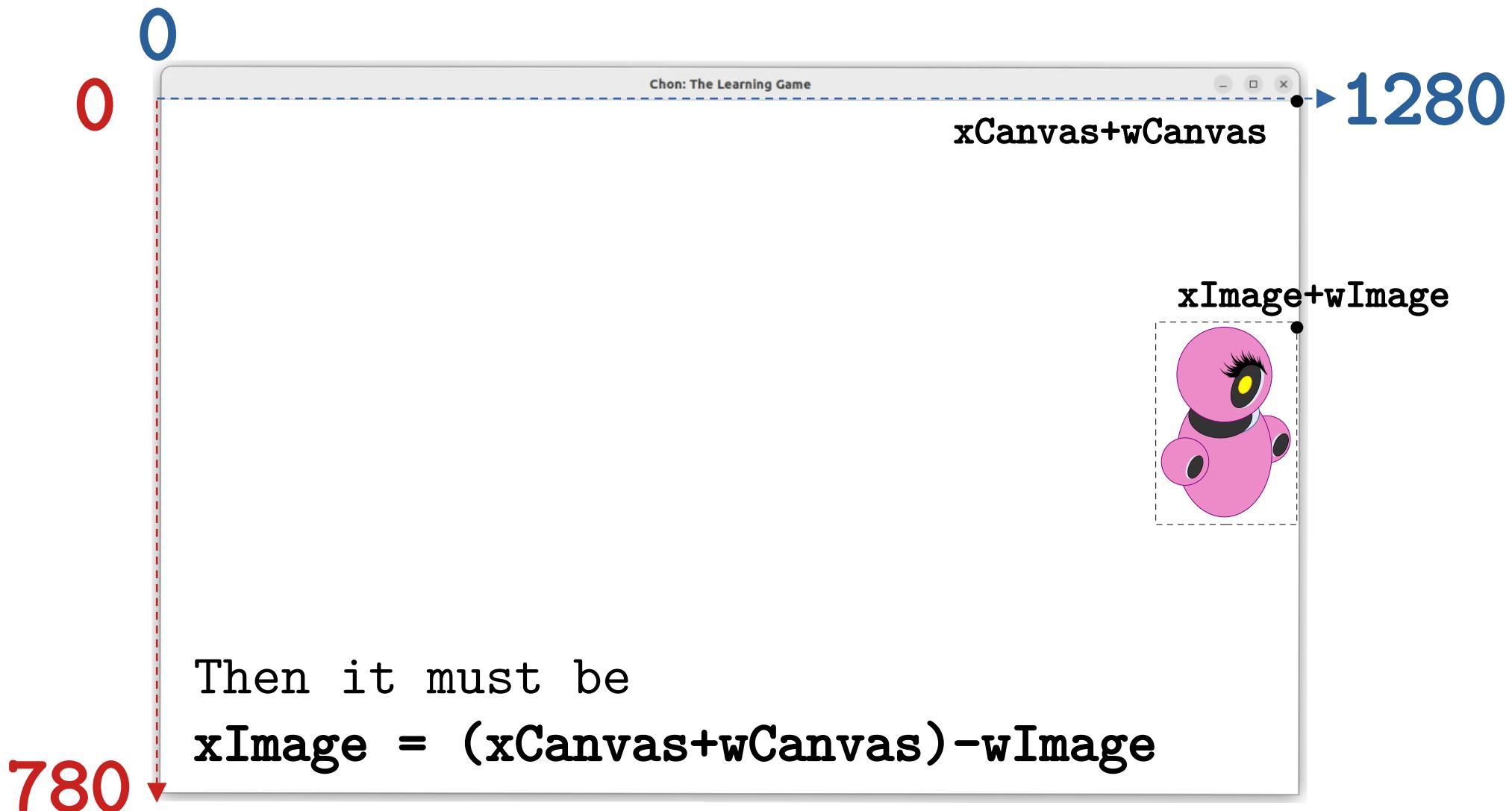
# Defining Boundaries at the RIGHT



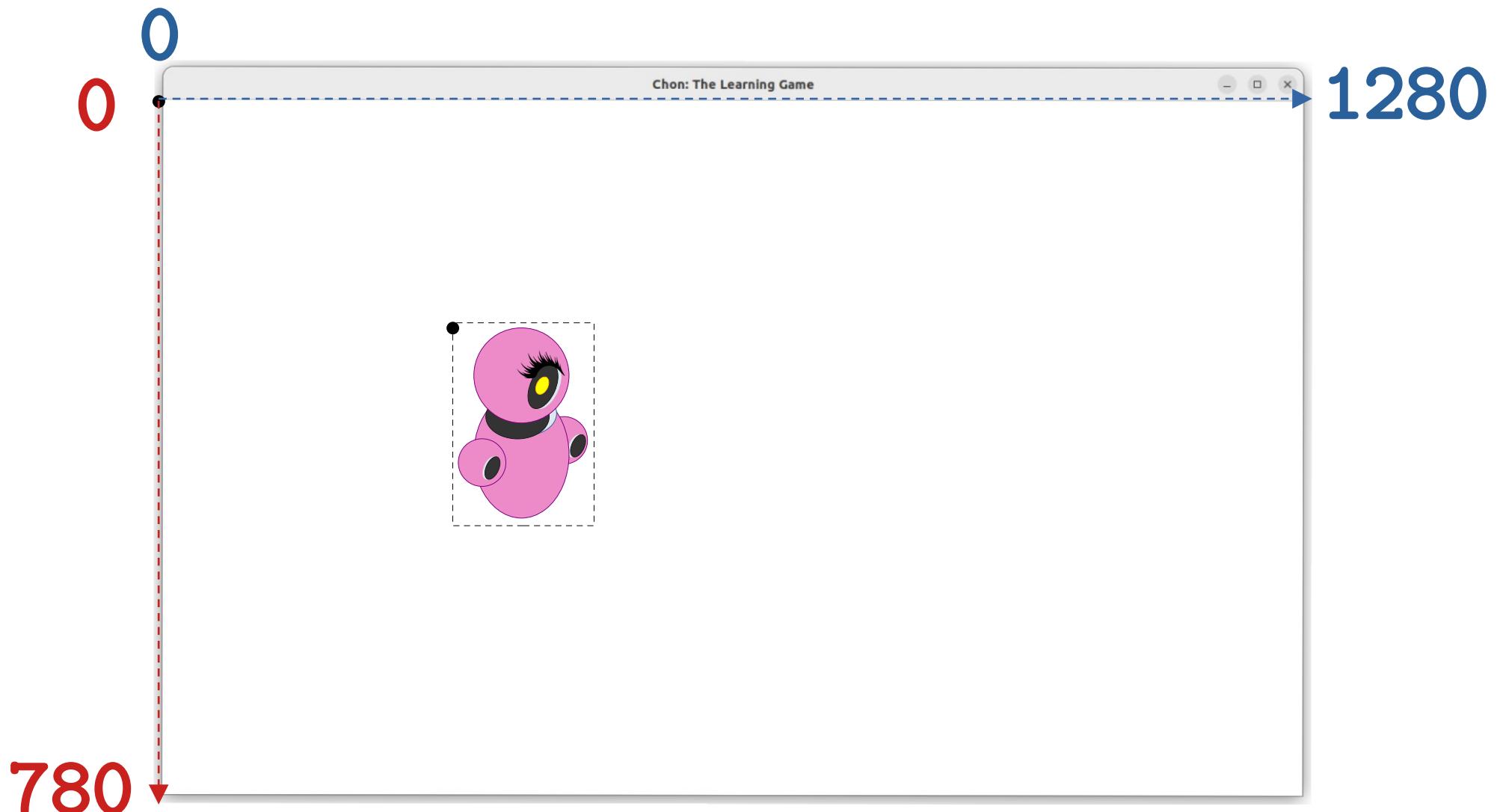
# Defining Boundaries at the RIGHT



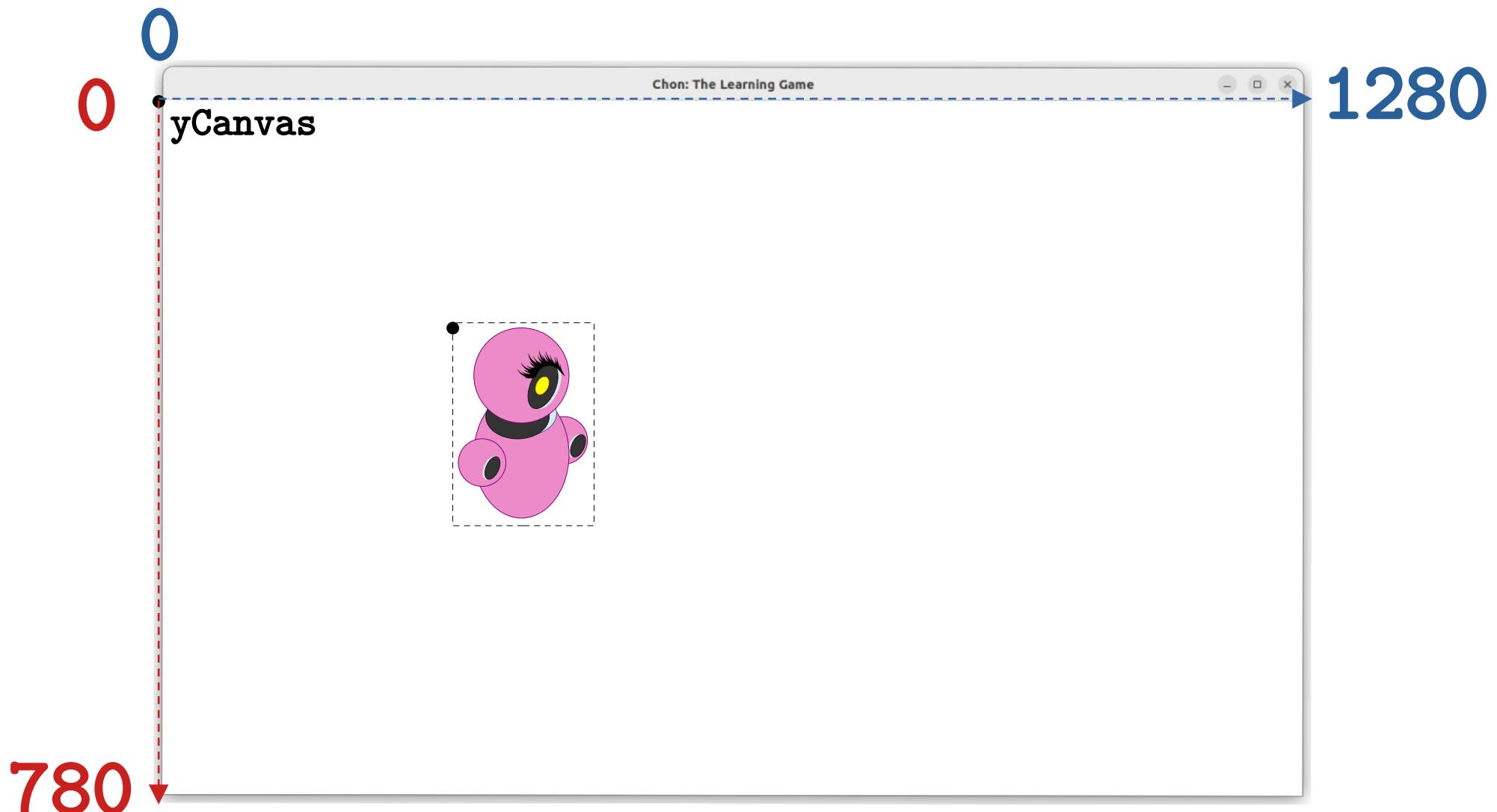
# Defining Boundaries at the RIGHT



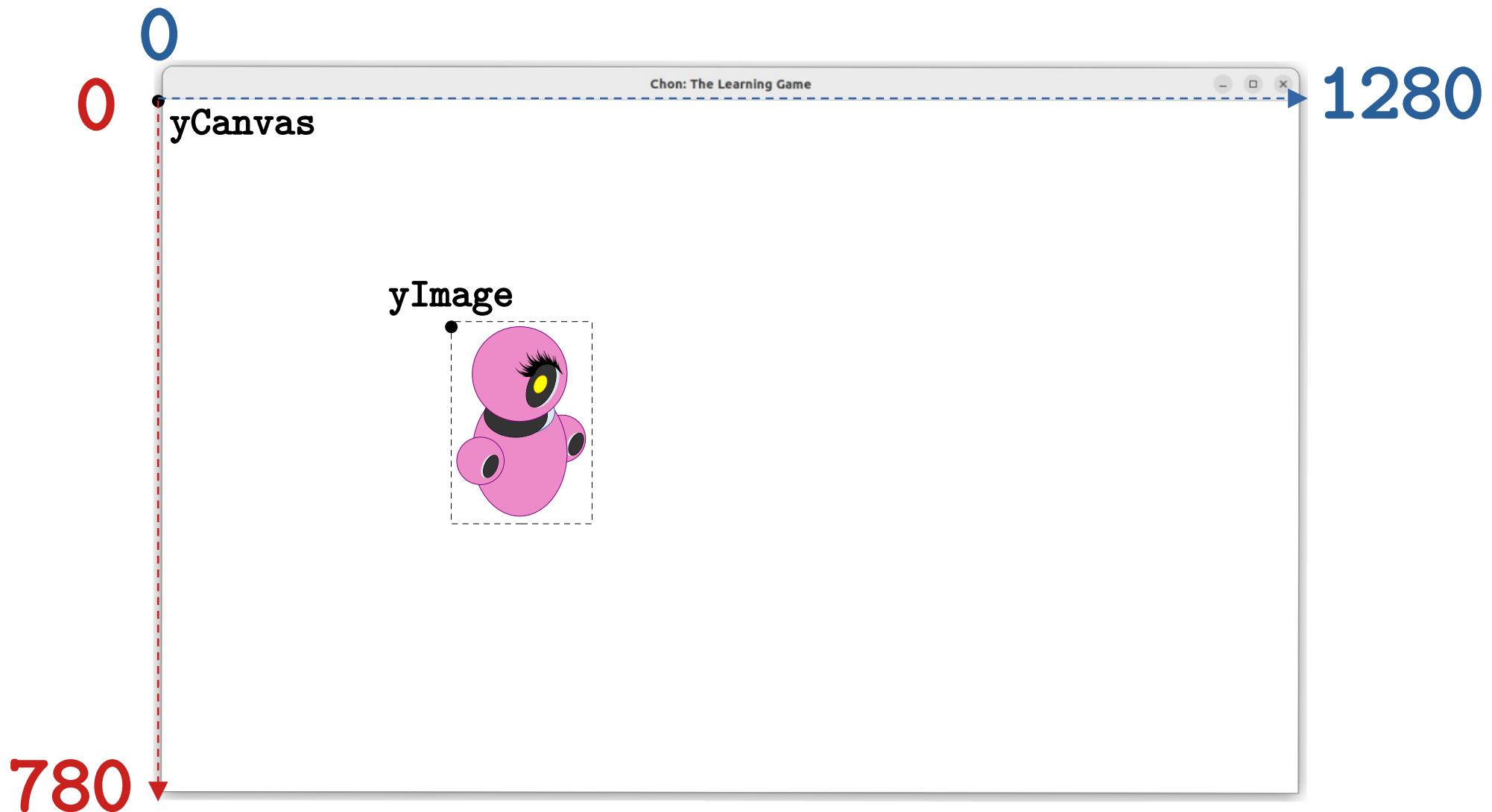
# Defining Boundaries at the TOP



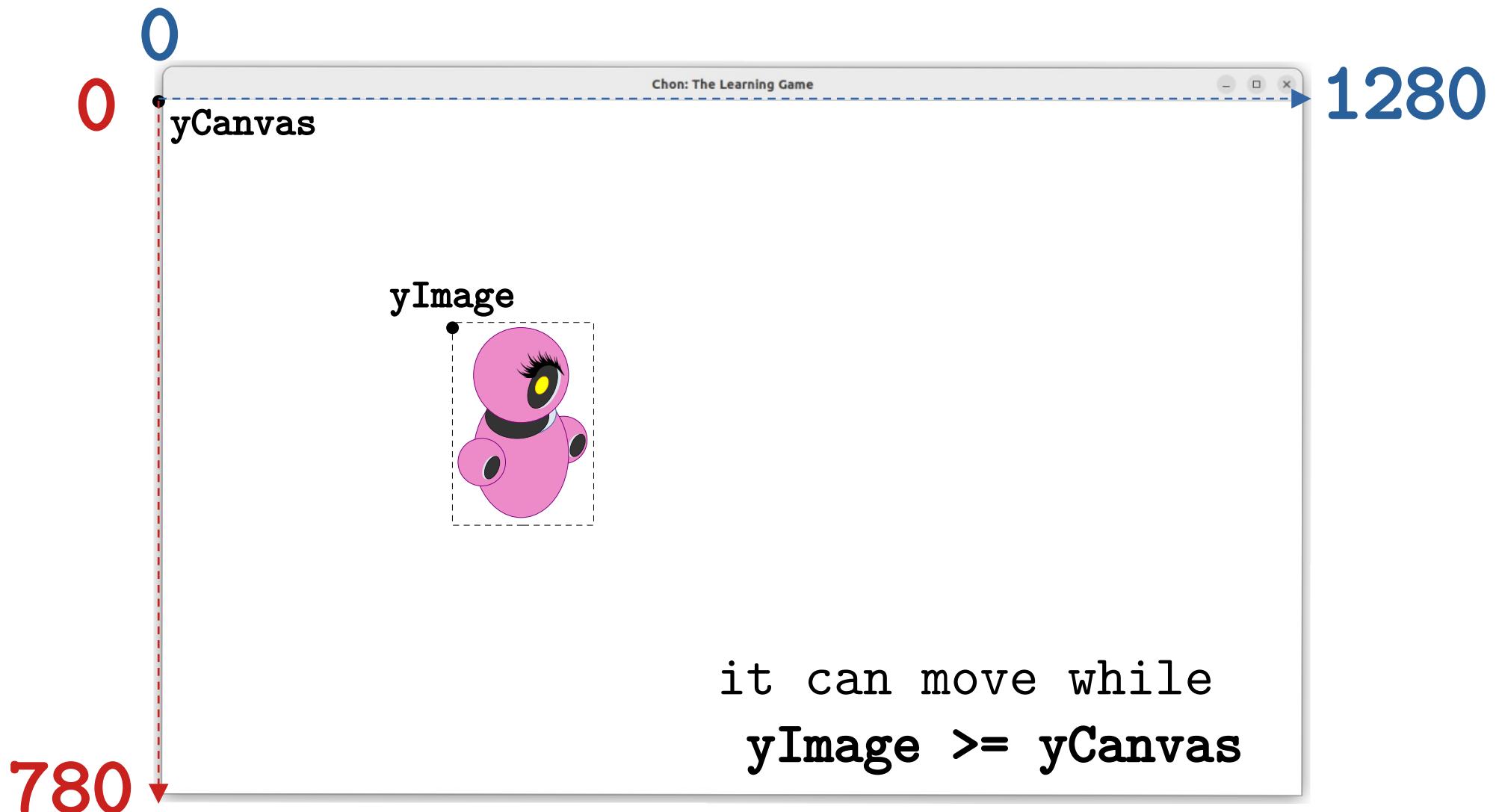
# Defining Boundaries at the TOP



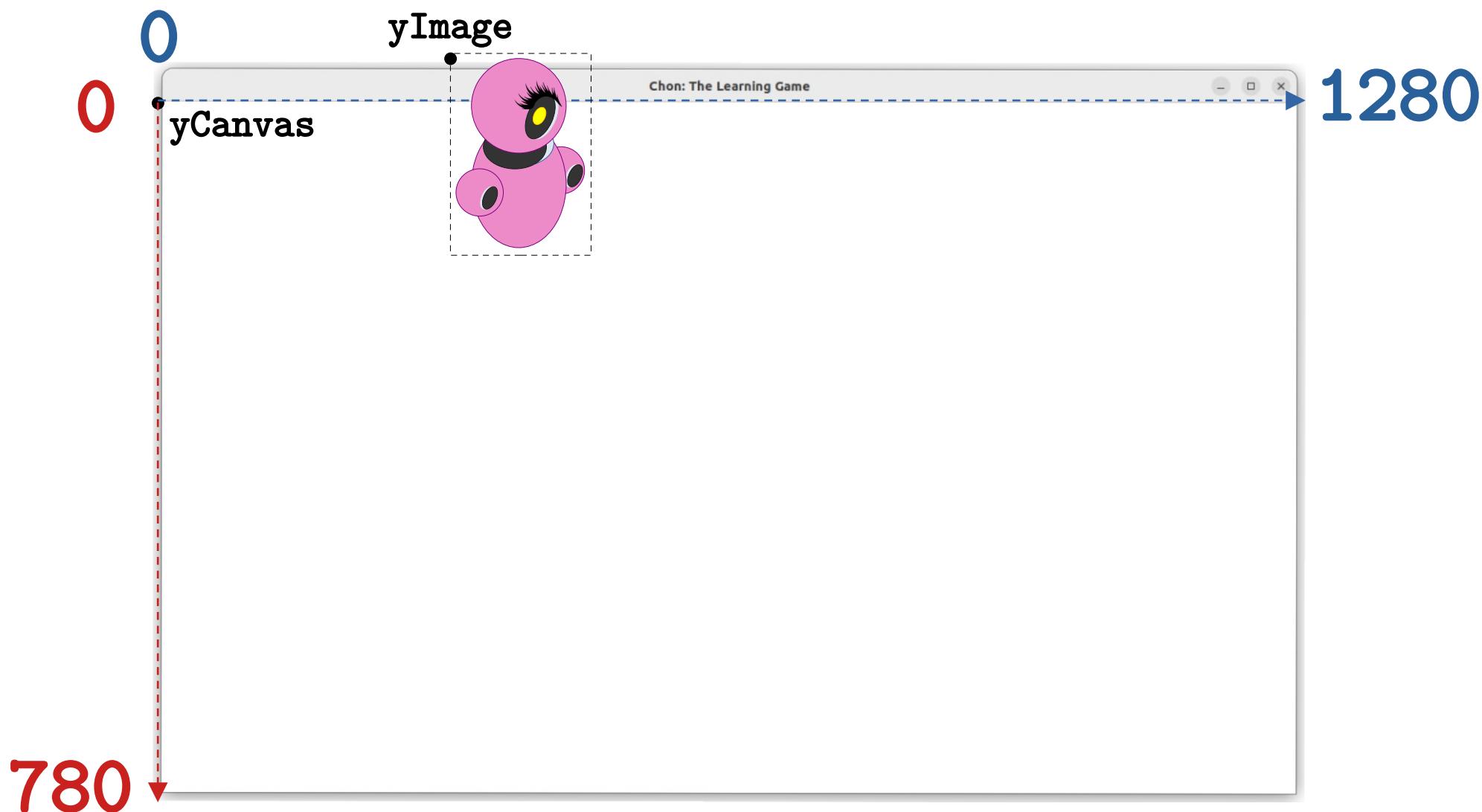
# Defining Boundaries at the TOP



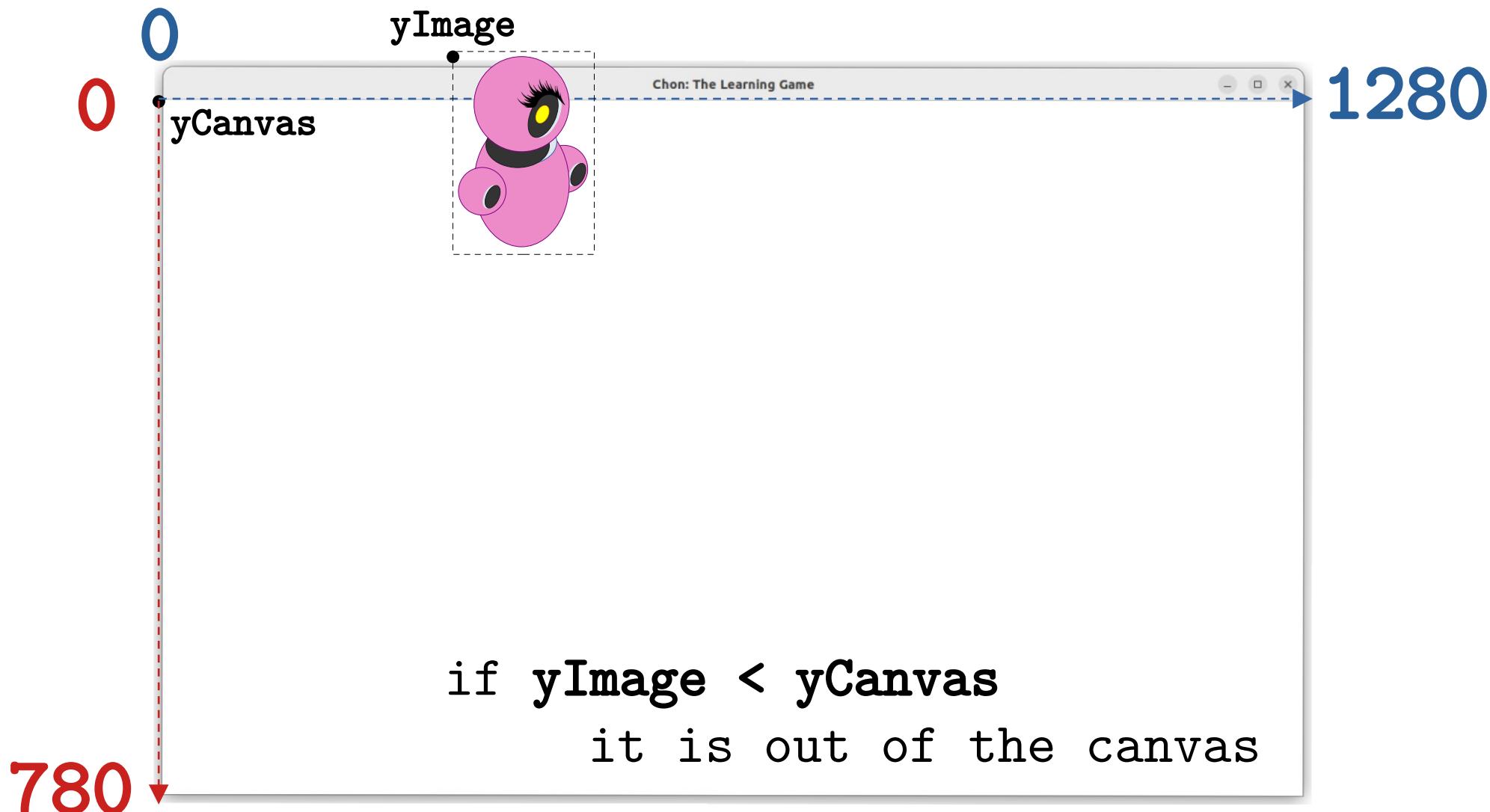
# Defining Boundaries at the TOP



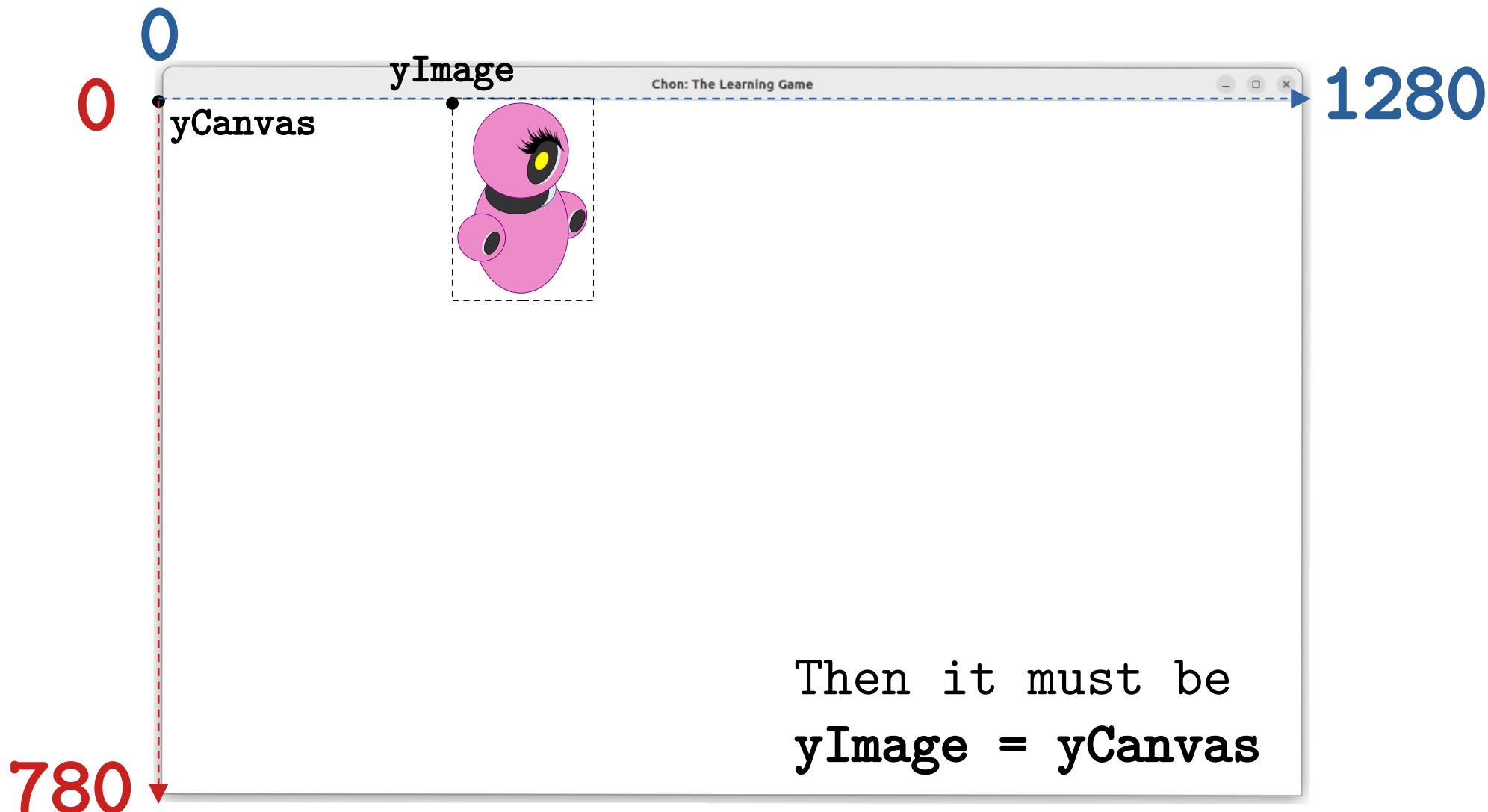
# Defining Boundaries at the TOP



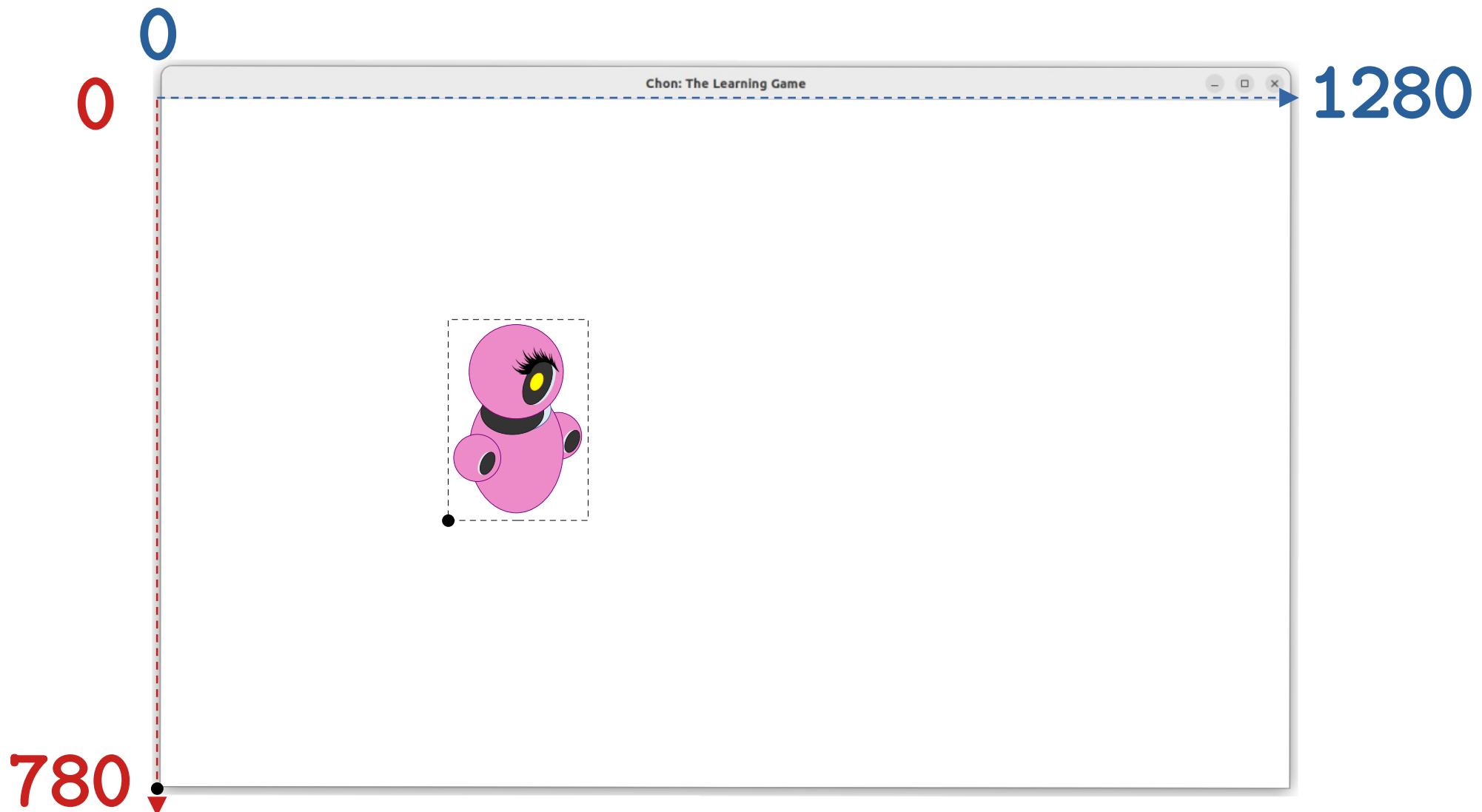
# Defining Boundaries at the TOP



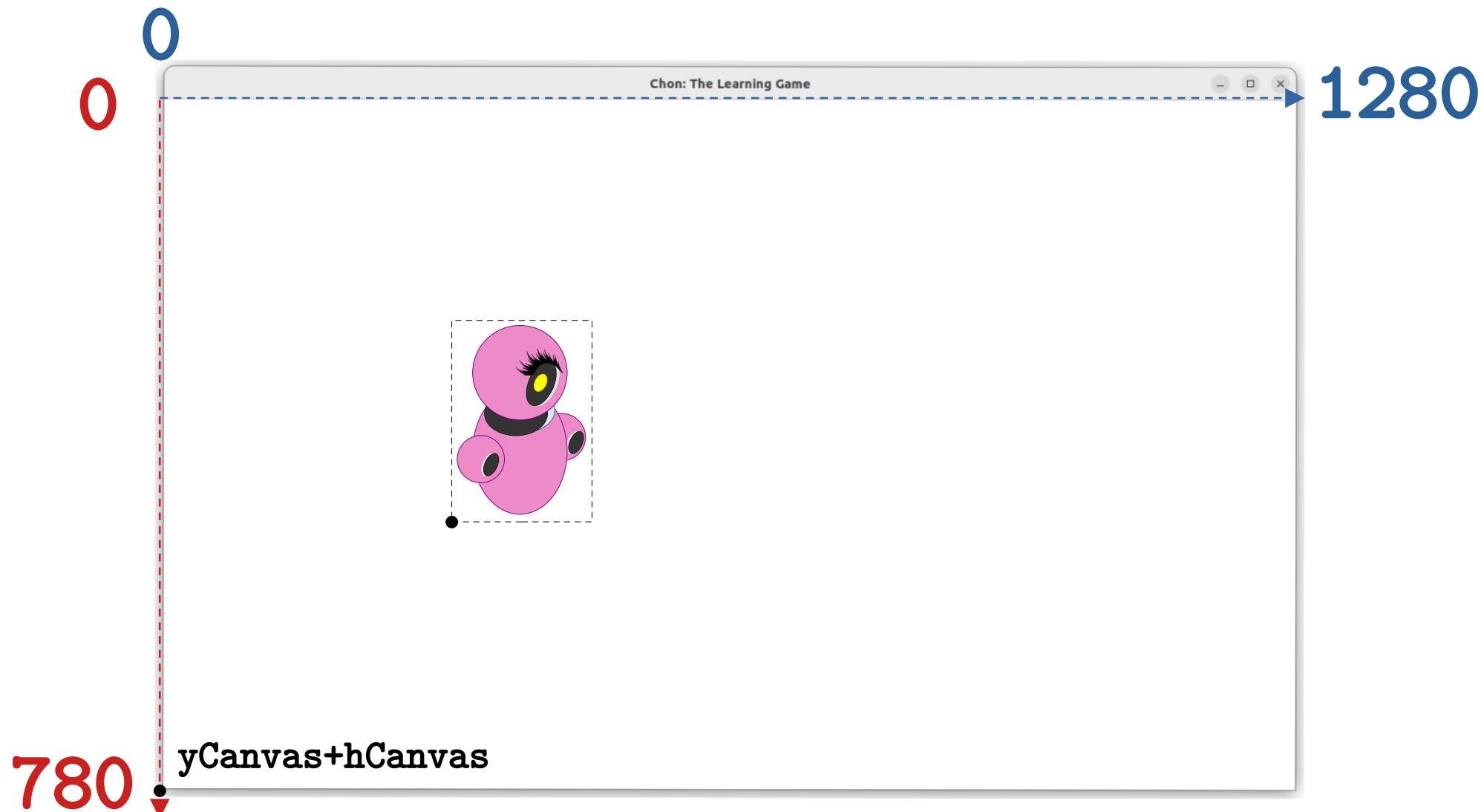
# Defining Boundaries at the TOP



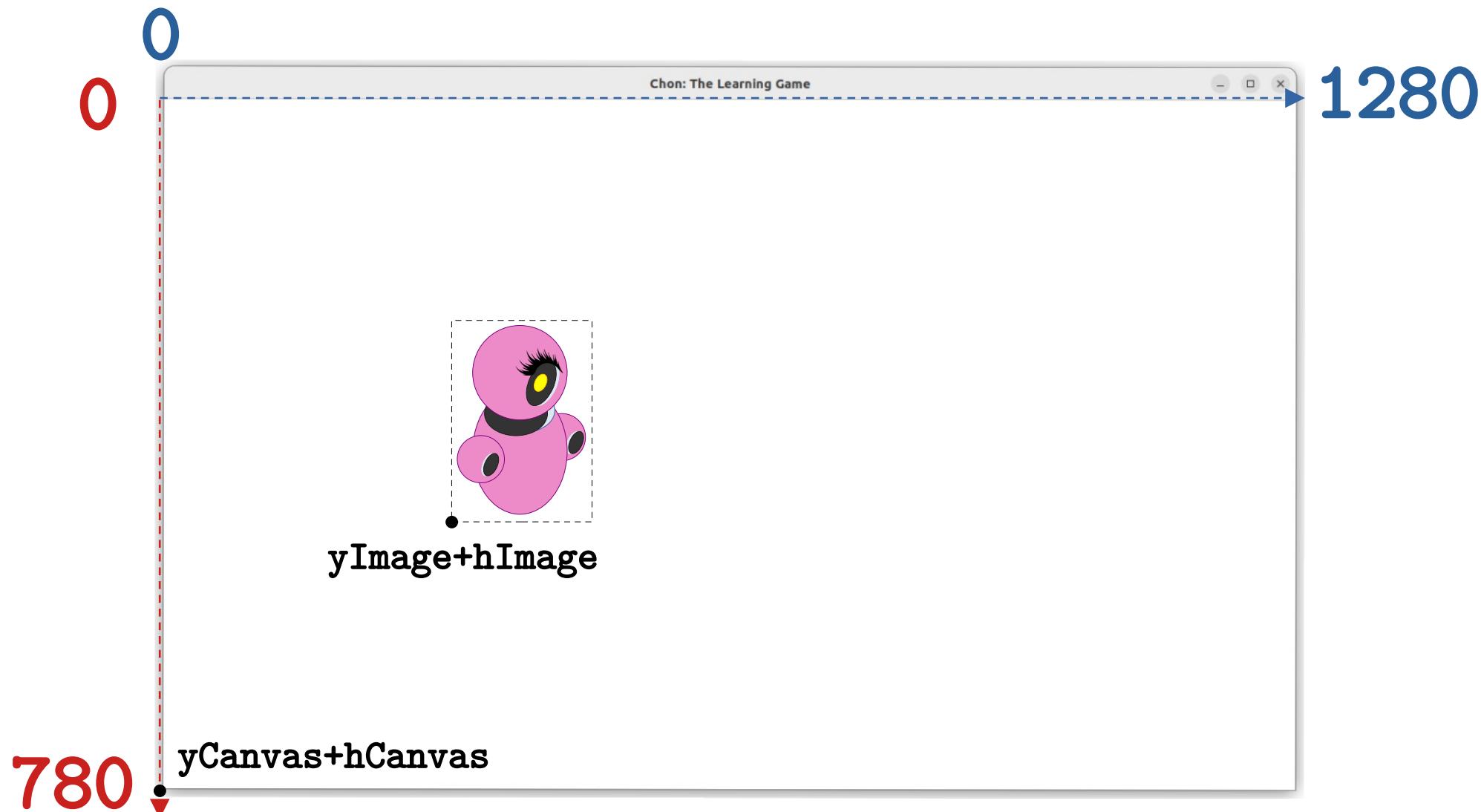
# Defining Boundaries at the BOTTOM



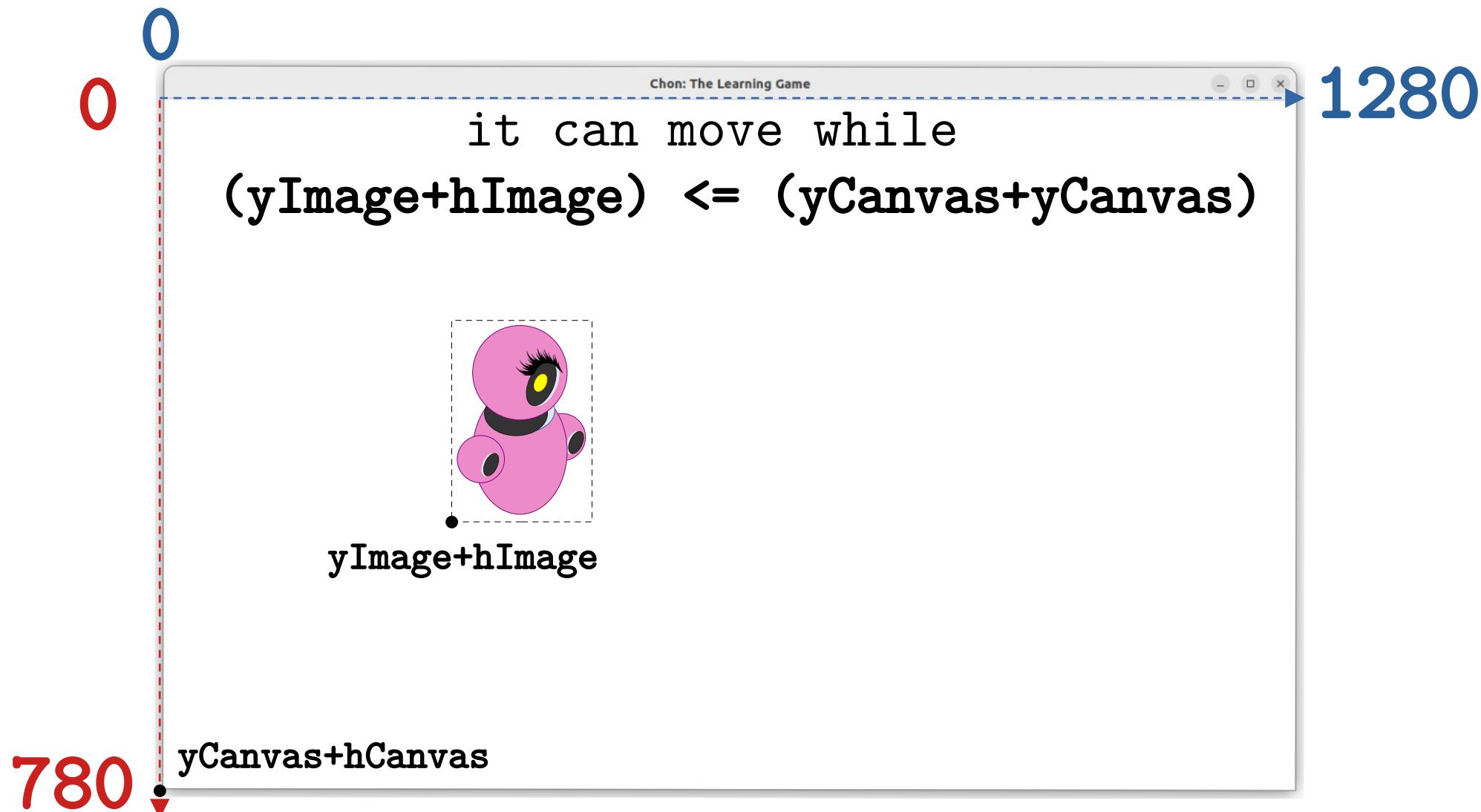
# Defining Boundaries at the BOTTOM



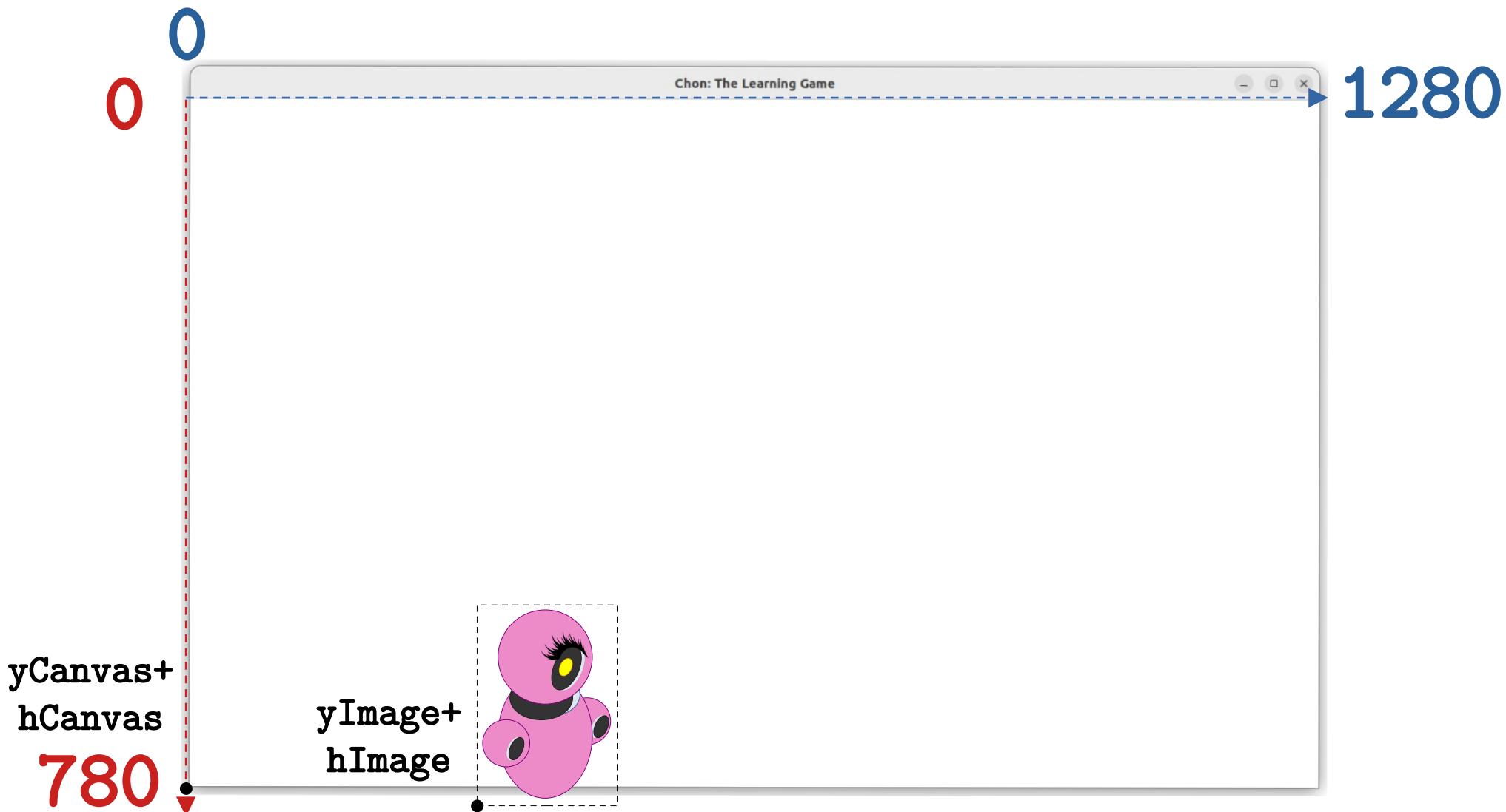
# Defining Boundaries at the BOTTOM



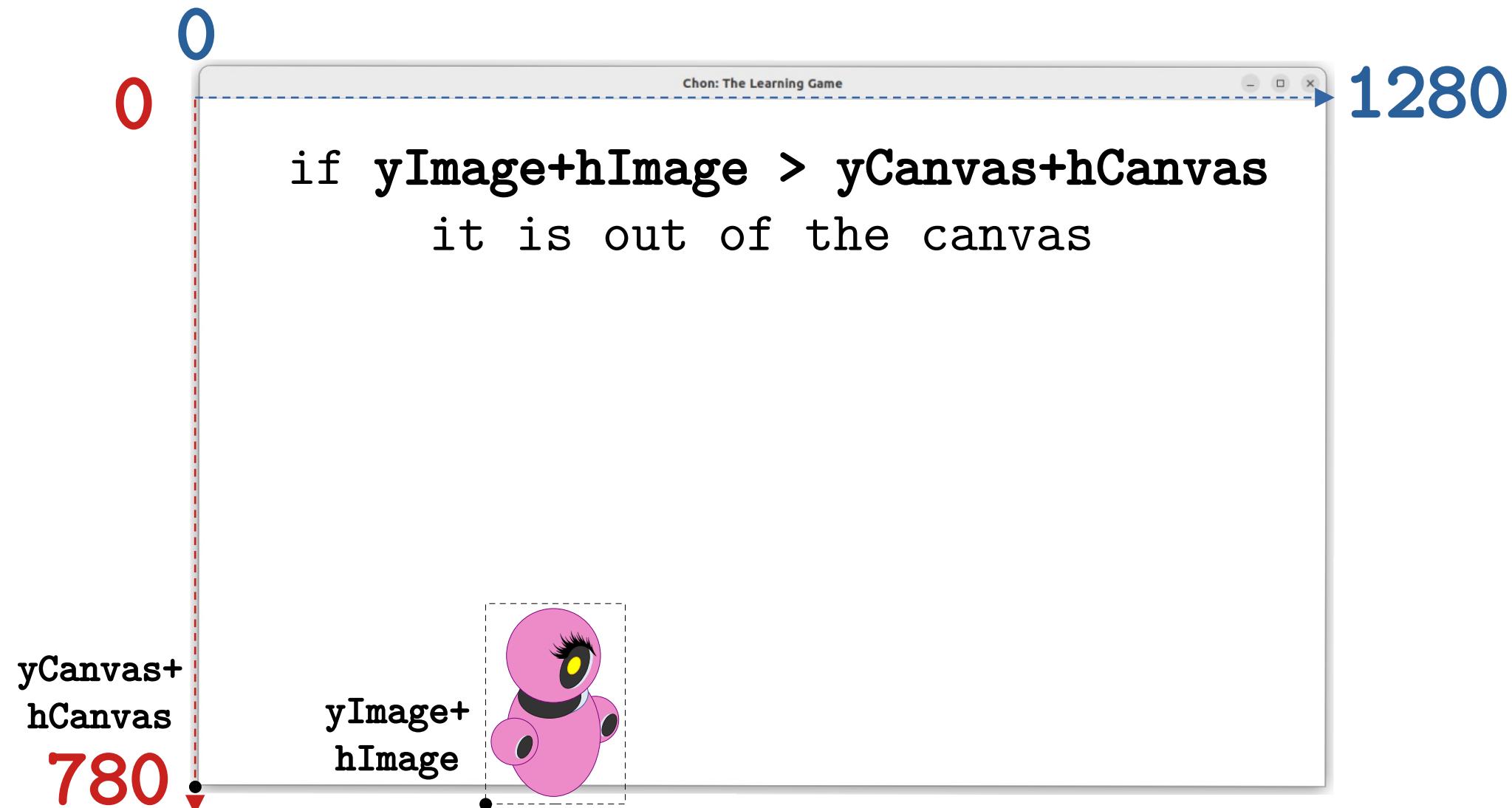
# Defining Boundaries at the BOTTOM



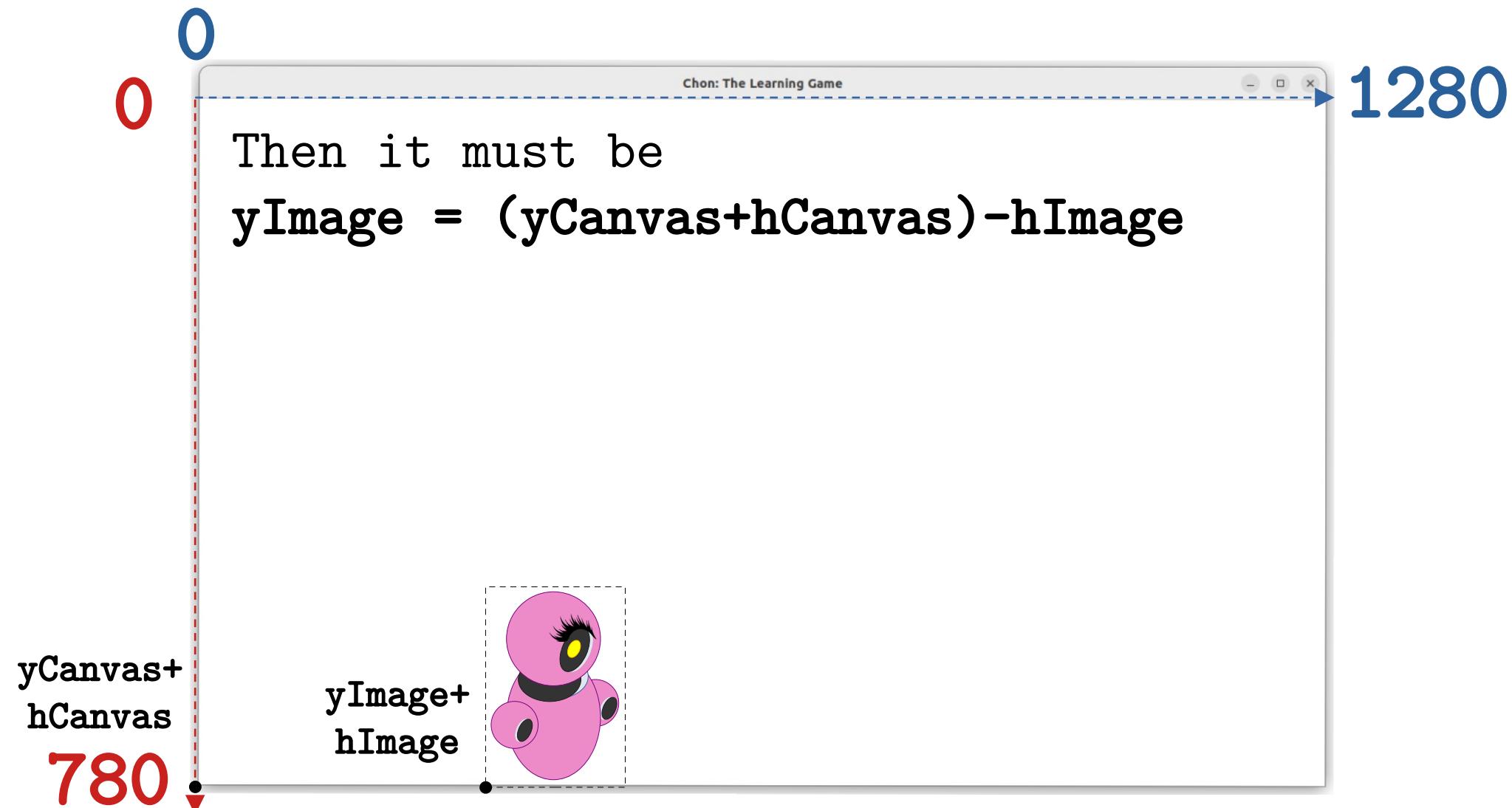
# Defining Boundaries at the BOTTOM



# Defining Boundaries at the BOTTOM



# Defining Boundaries at the BOTTOM



# Printing a Status Panel



# Printing a Status Panel



# Image Overlapping

The order in which you place the **Image** matters.

# Image Overlapping

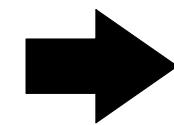
The order in which you place the **Image** matters.

```
gc.drawImage(chonBot, x:920, y:440, wImageBot, hImageBot);  
gc.drawImage(chonBota, x:400, y:390, wImageBot, hImageBot);
```

# Image Overlapping

The order in which you place the **Image** matters.

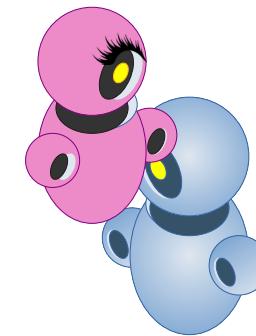
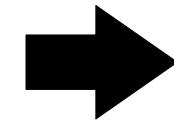
```
gc.drawImage(chonBot, x:920, y:440, wImageBot, hImageBot);  
gc.drawImage(chonBota, x:400, y:390, wImageBot, hImageBot);
```



# Image Overlapping

The order in which you place the **Image** matters.

```
gc.drawImage(chonBot, x:920, y:440, wImageBot, hImageBot);  
gc.drawImage(chonBota, x:400, y:390, wImageBot, hImageBot);
```



# Image Overlapping

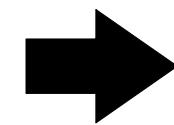
The order in which you place the **Image** matters.

```
gc.drawImage(chonBota, x:400, y:390, wImageBot, hImageBot);
gc.drawImage(chonBot, x:920, y:440, wImageBot, hImageBot);
```

# Image Overlapping

The order in which you place the **Image** matters.

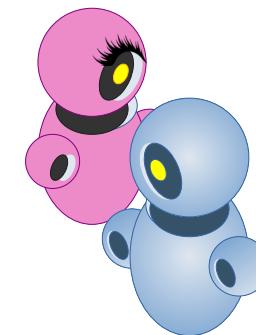
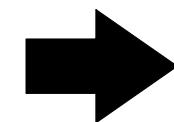
```
gc.drawImage(chonBota, x:400, y:390, wImageBot, hImageBot);  
gc.drawImage(chonBot, x:920, y:440, wImageBot, hImageBot);
```



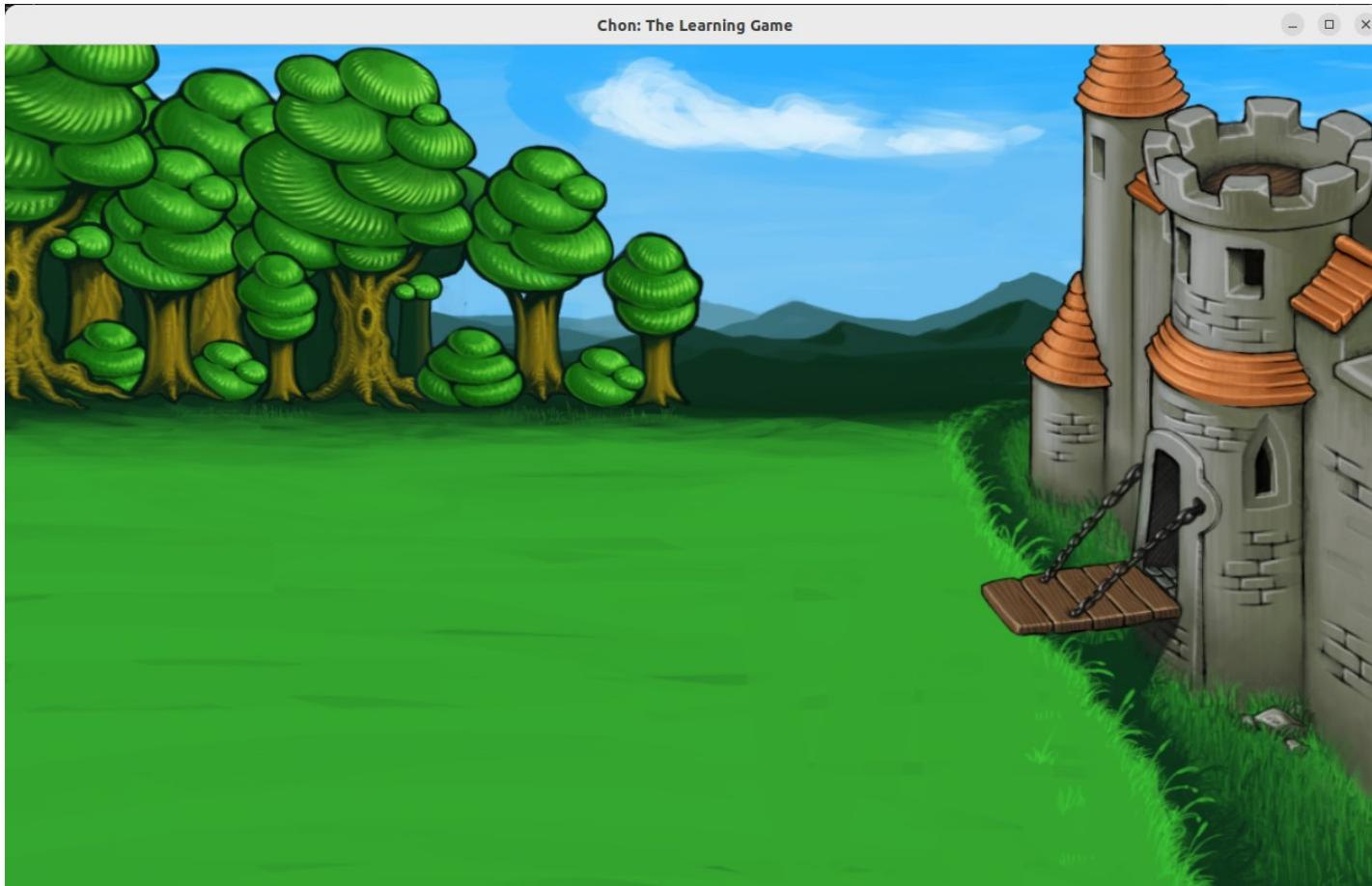
# Image Overlapping

The order in which you place the **Image** matters.

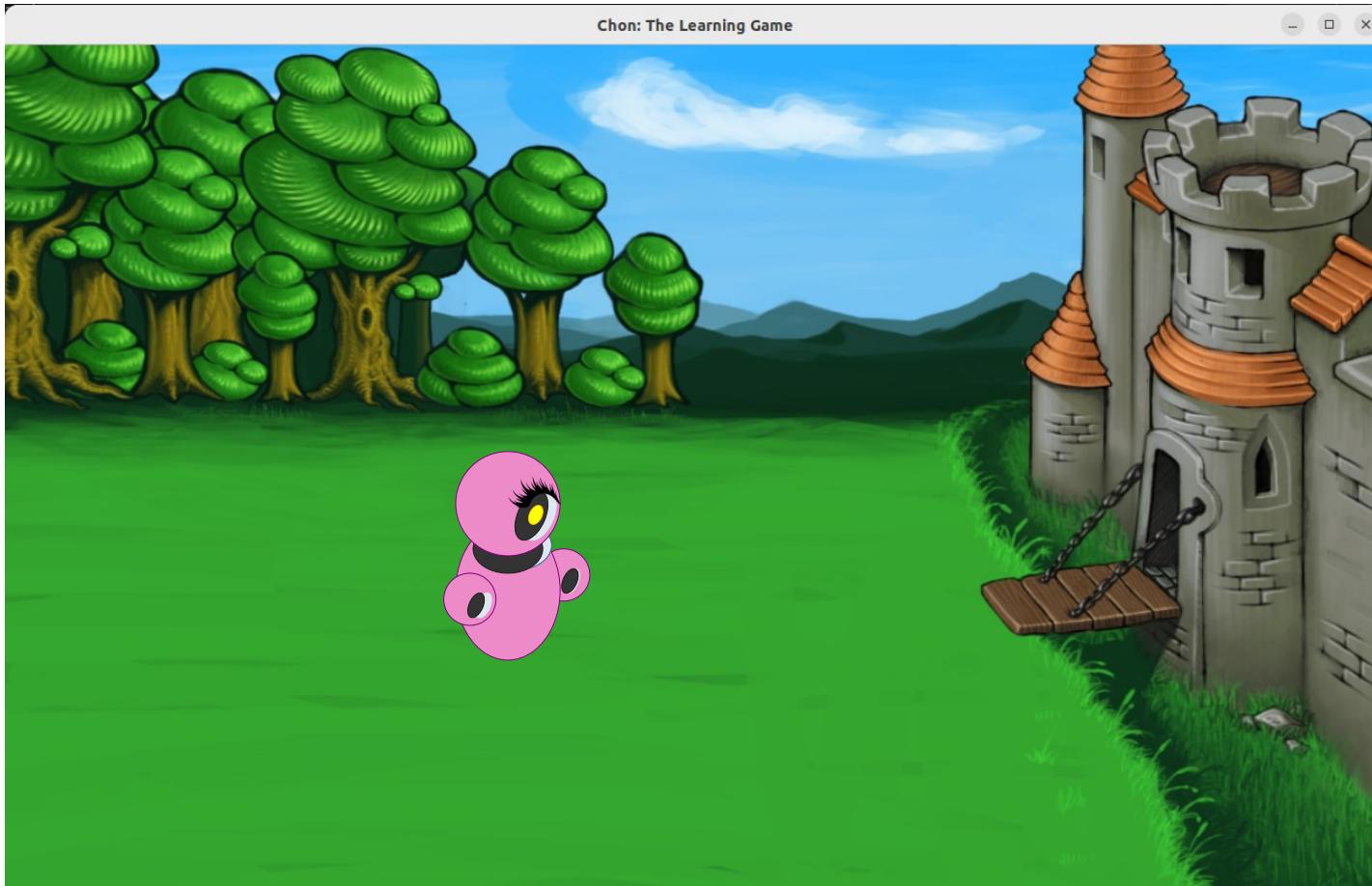
```
gc.drawImage(chonBota, x:400, y:390, wImageBot, hImageBot);  
gc.drawImage(chonBot, x:920, y:440, wImageBot, hImageBot);
```



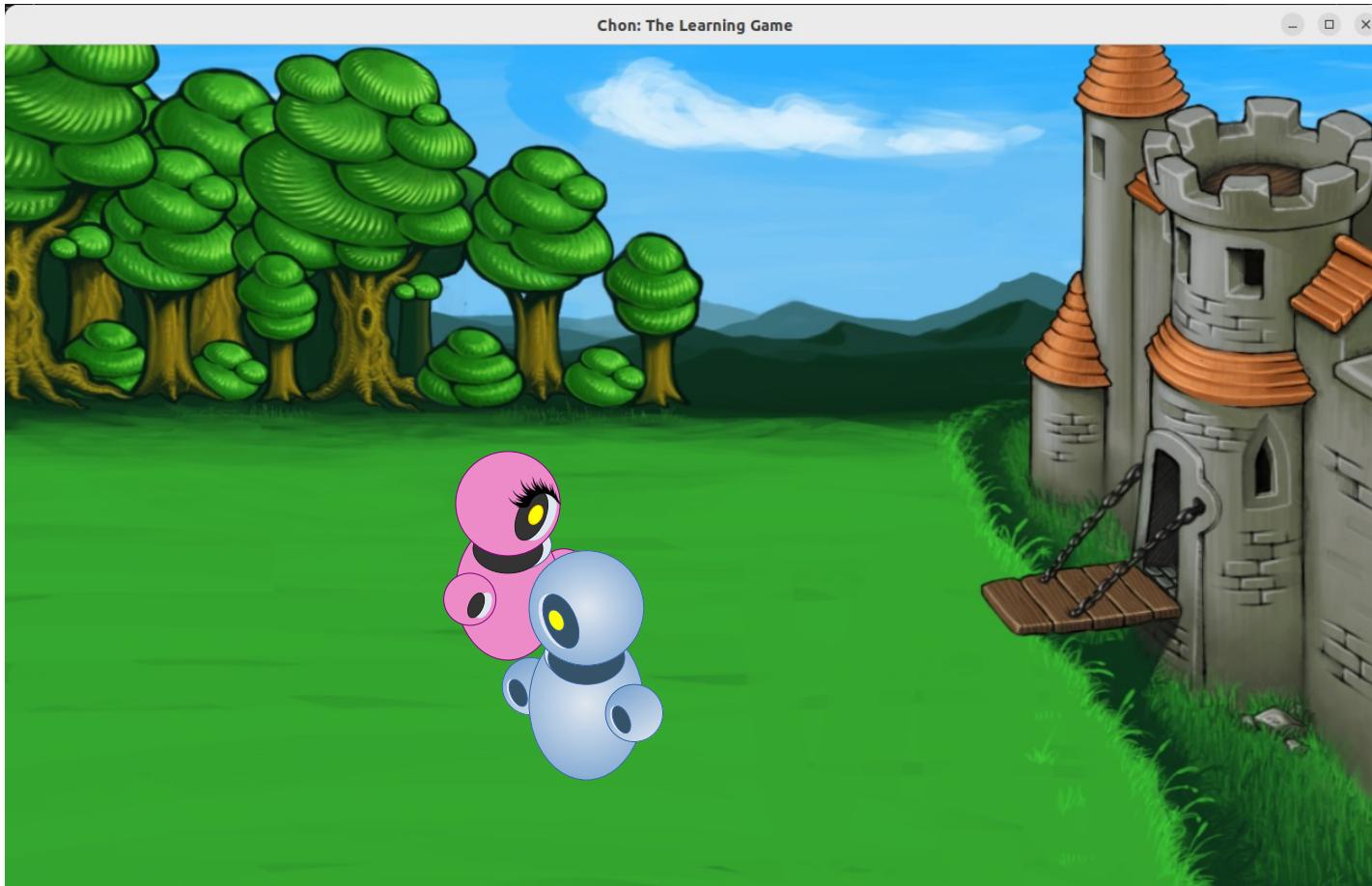
# Image Overlapping



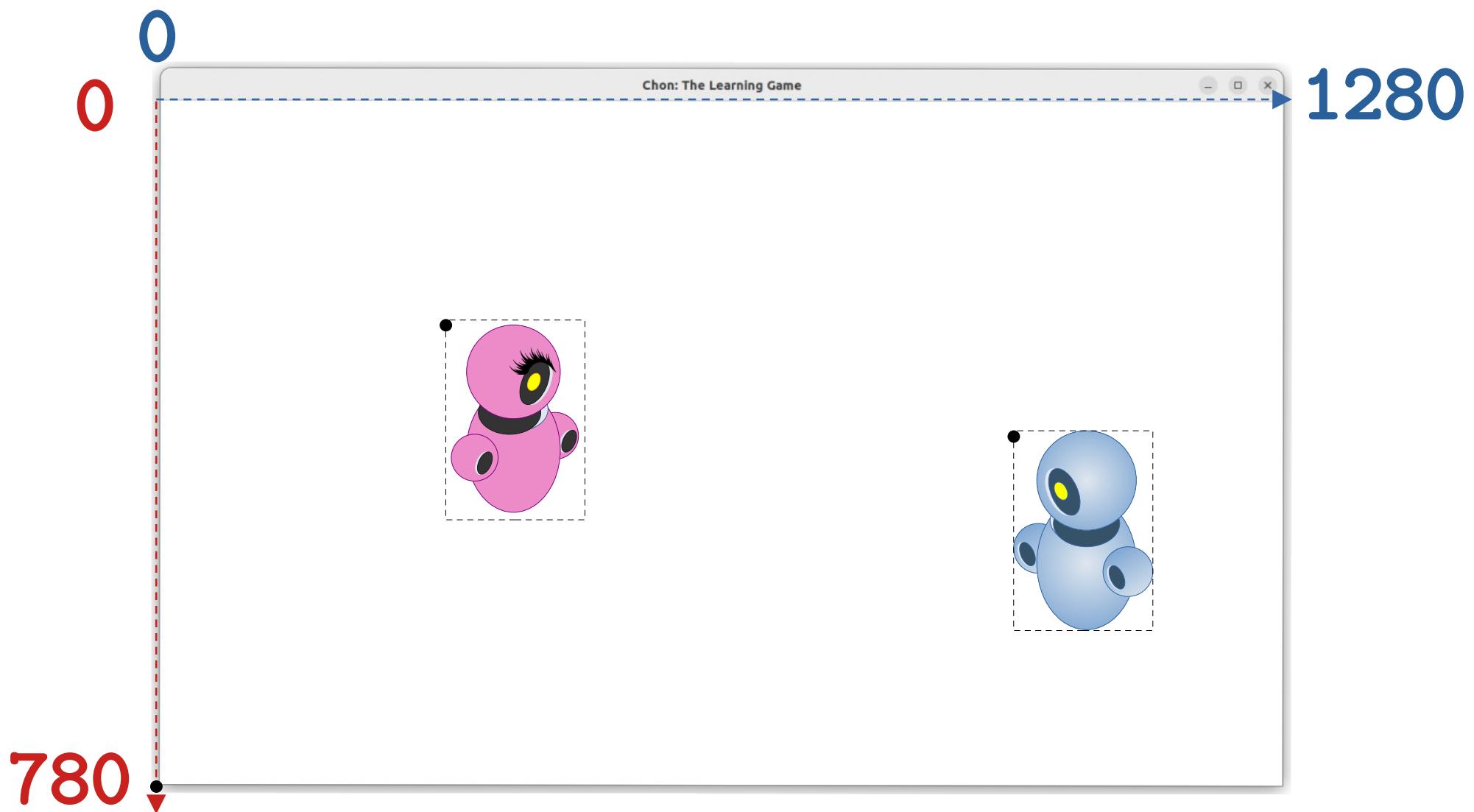
# Image Overlapping



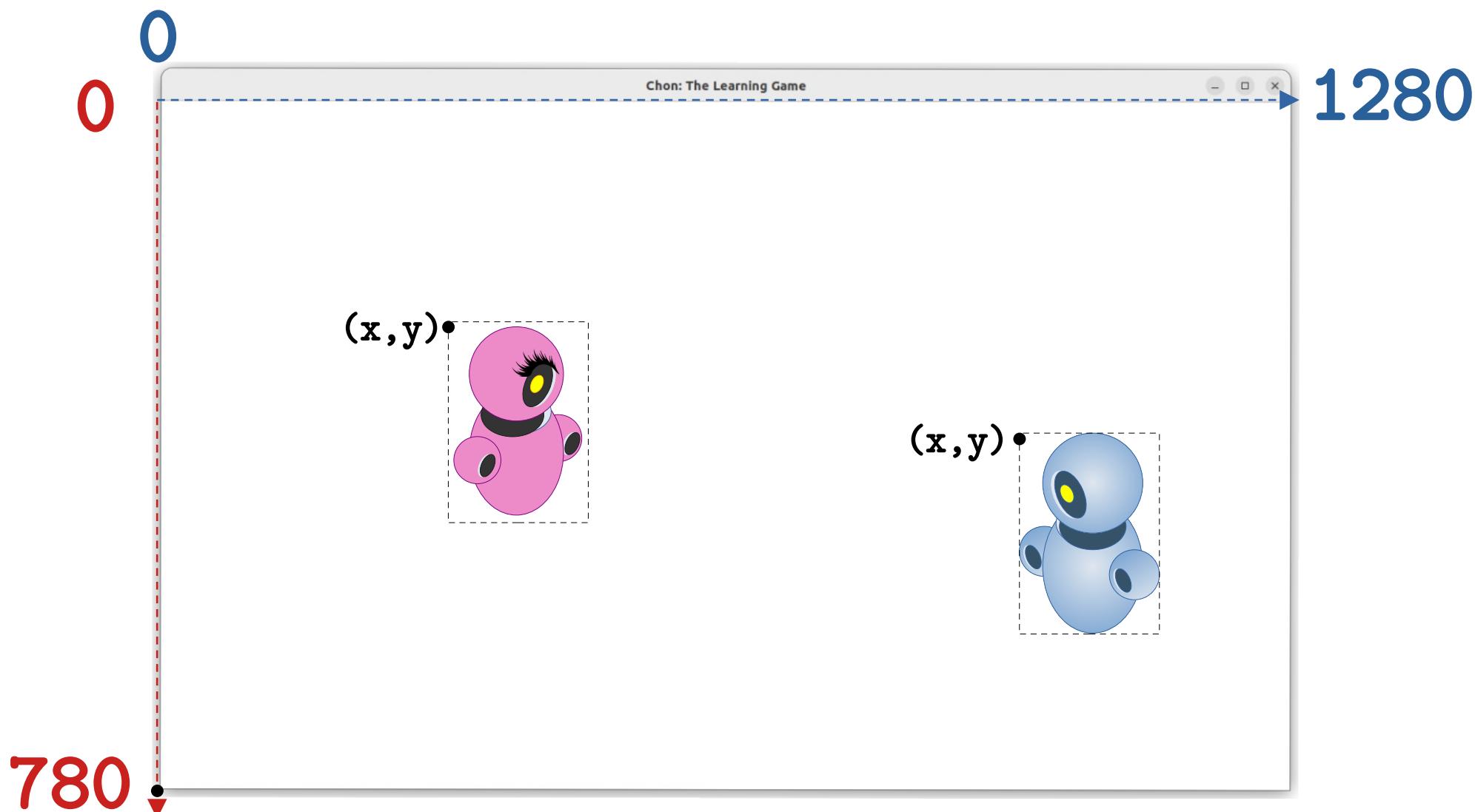
# Image Overlapping



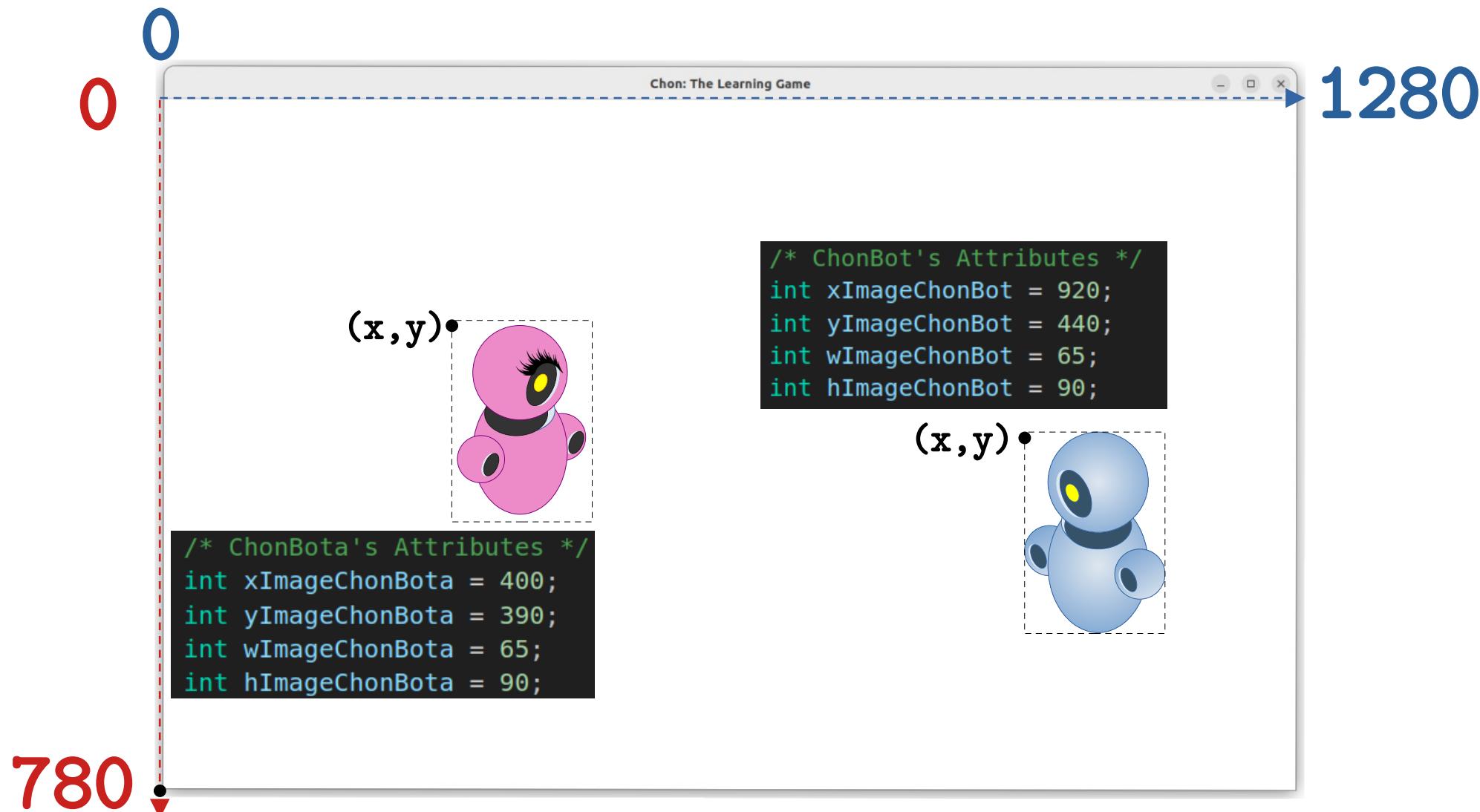
# Moving Another Object



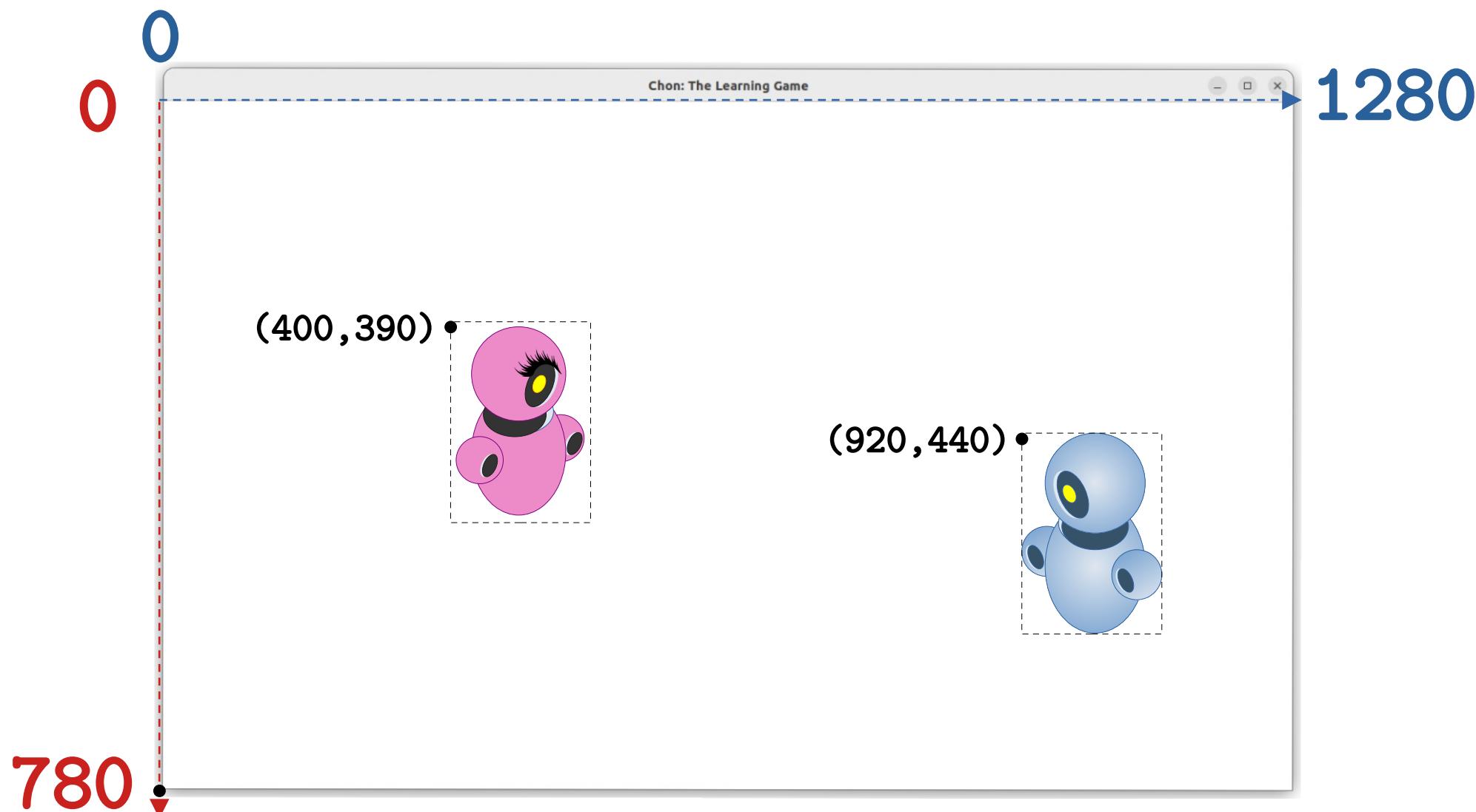
# Moving Another Object



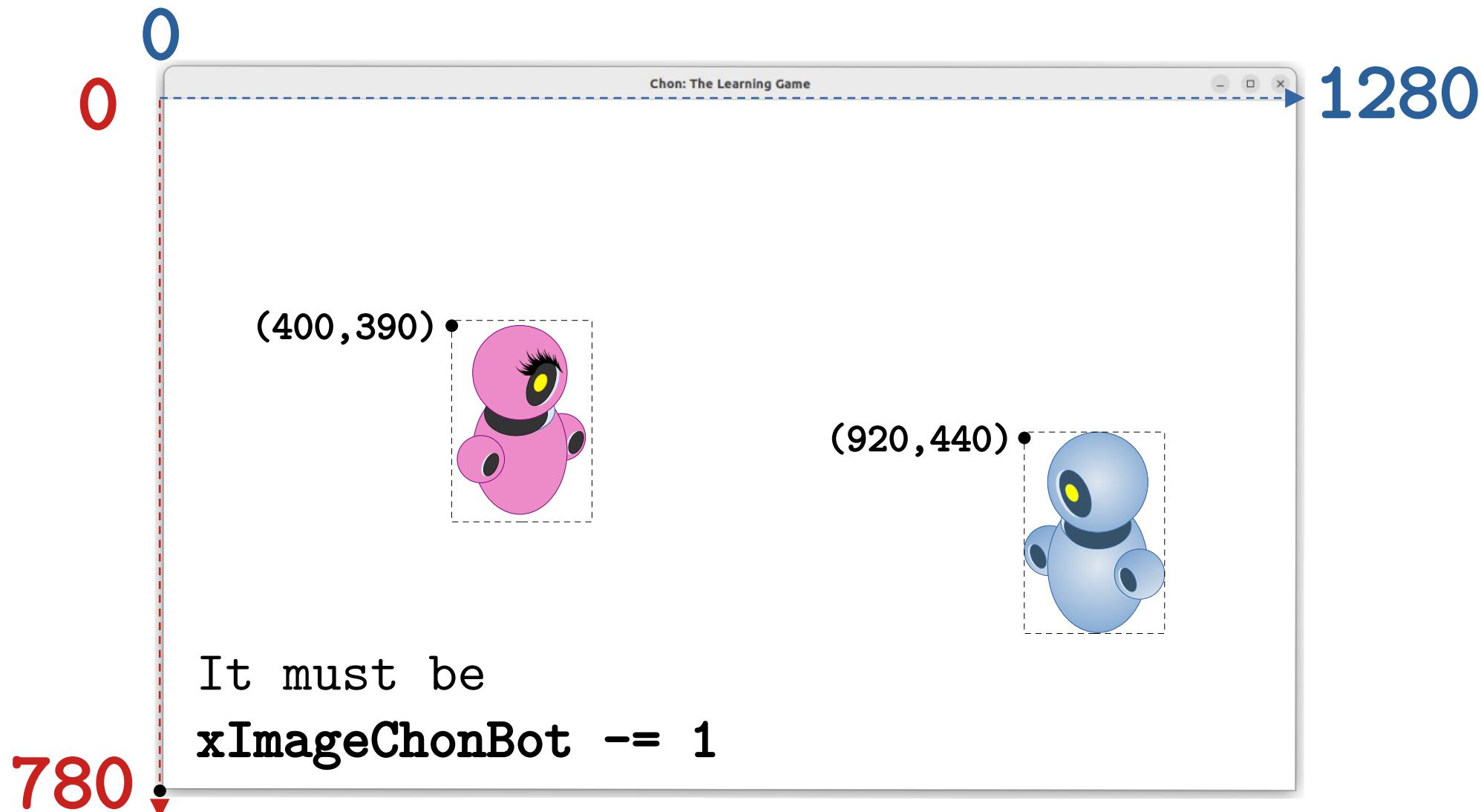
# Moving Another Object



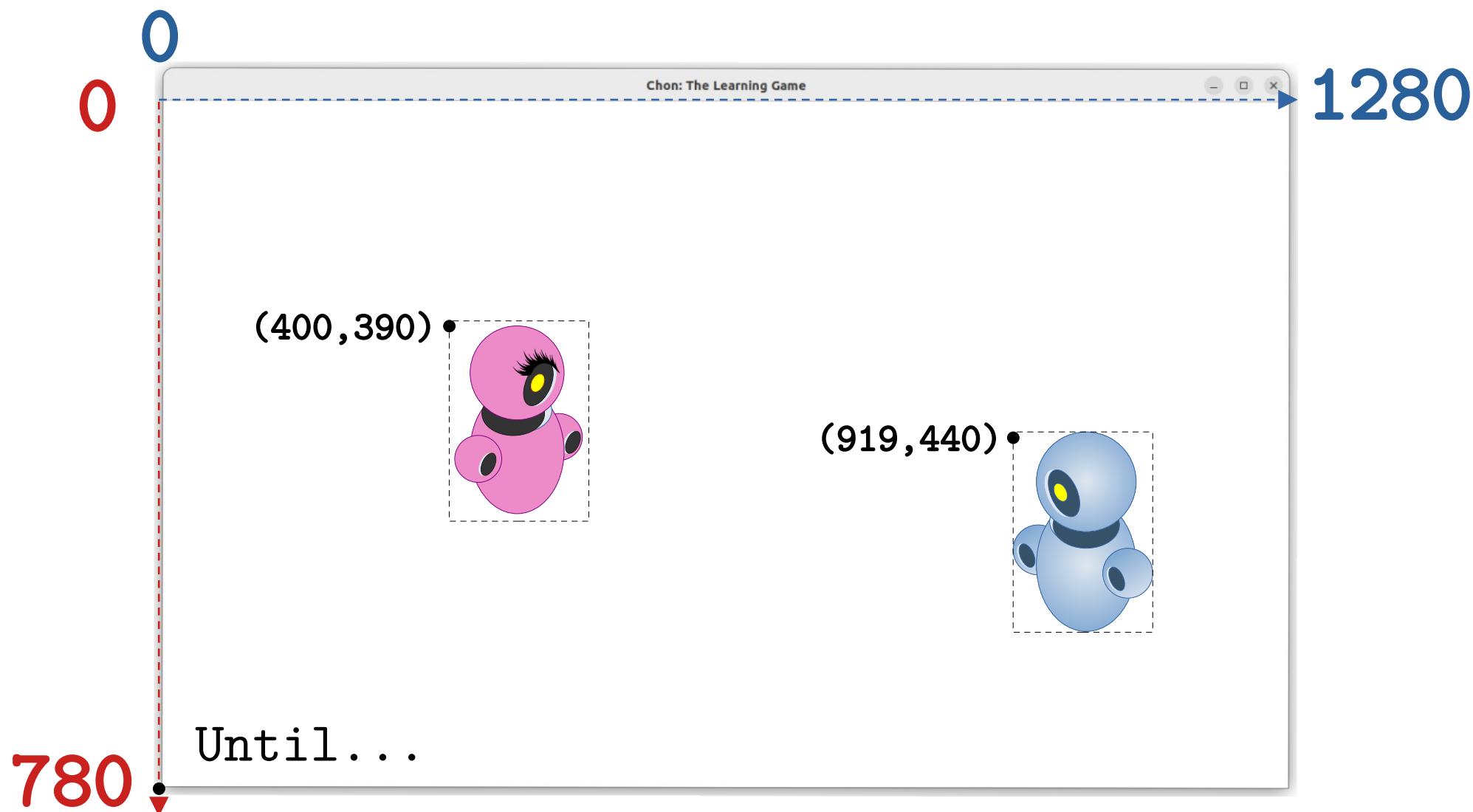
# Moving Another Object to the LEFT



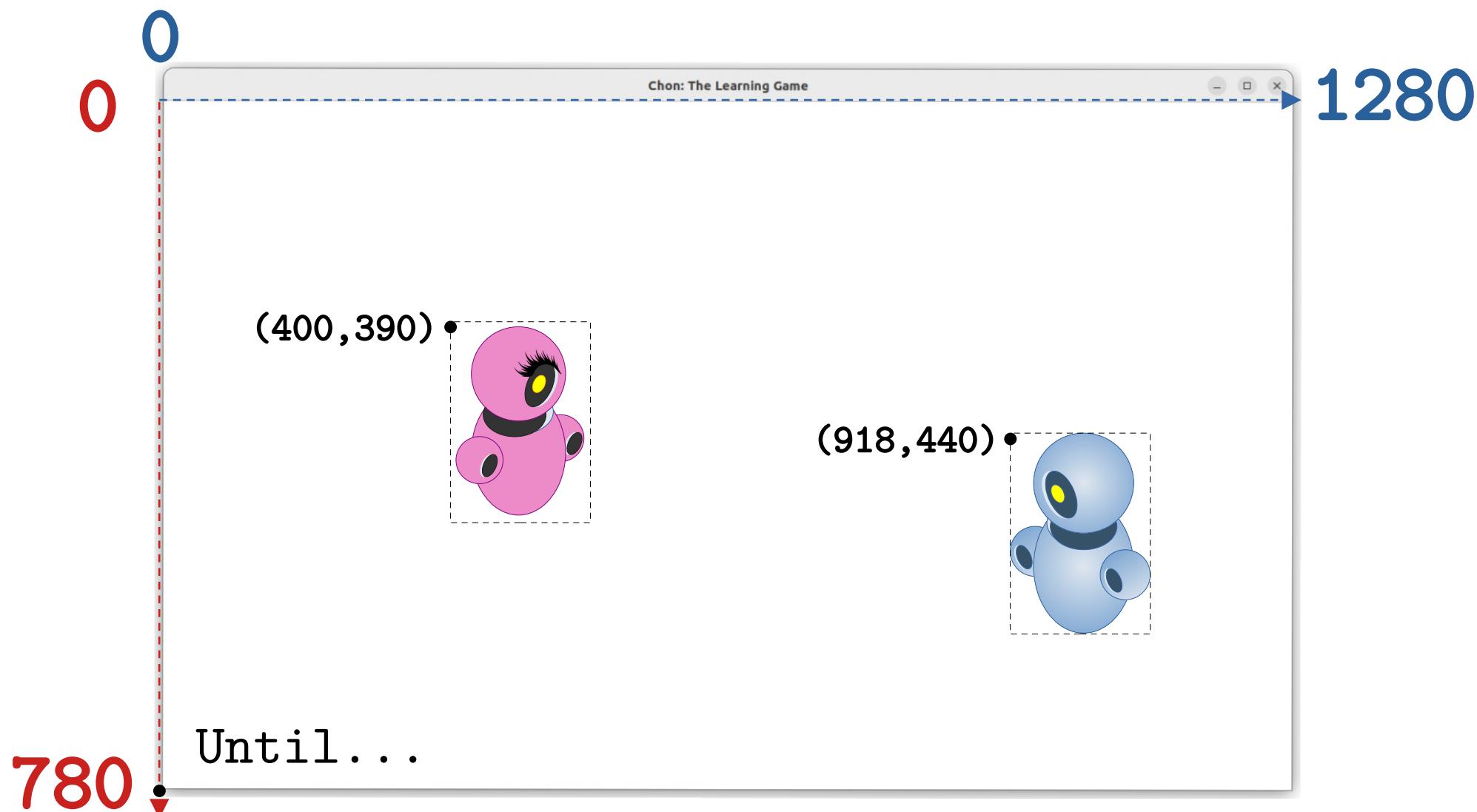
# Moving Another Object to the LEFT



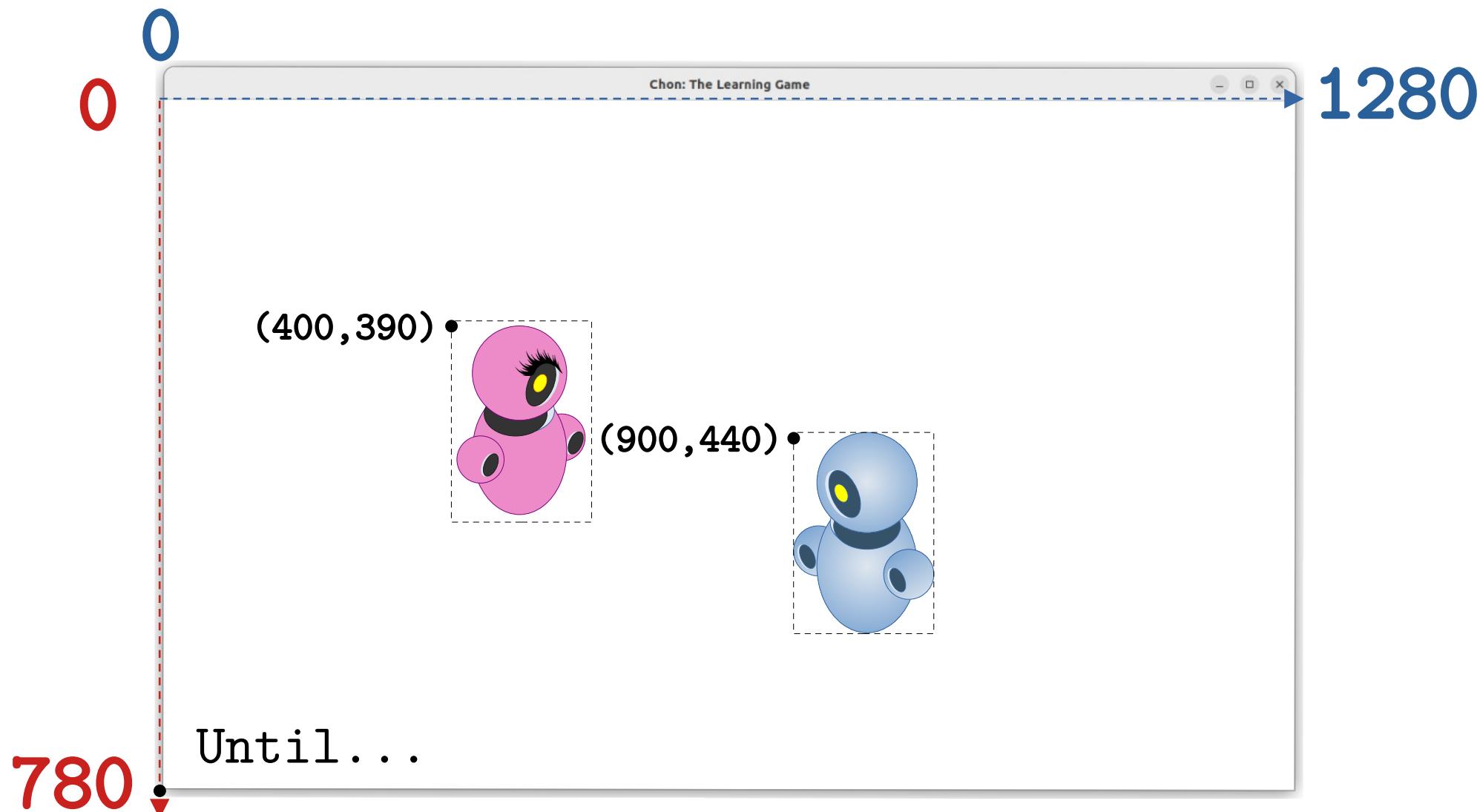
# Moving Another Object to the LEFT



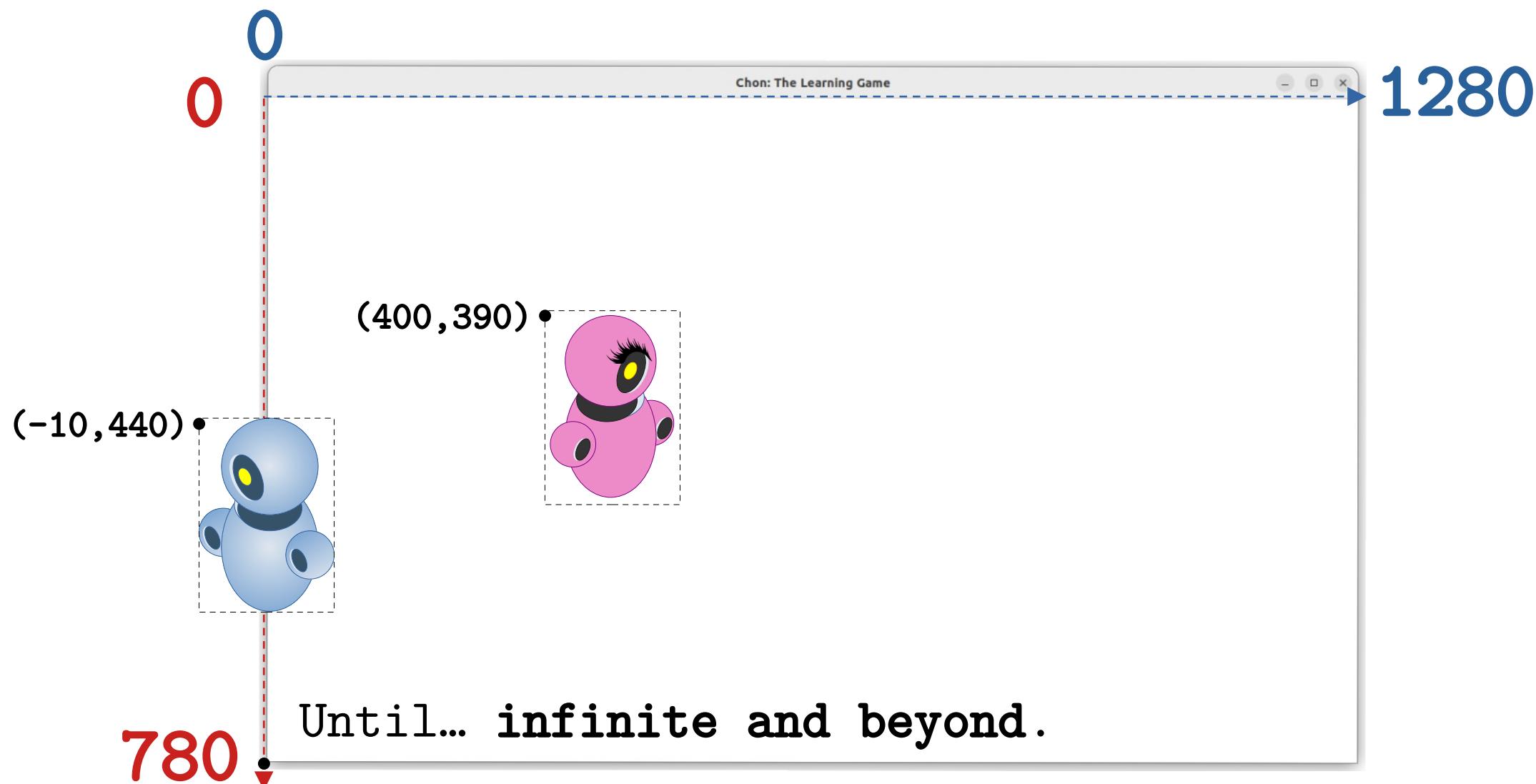
# Moving Another Object to the LEFT



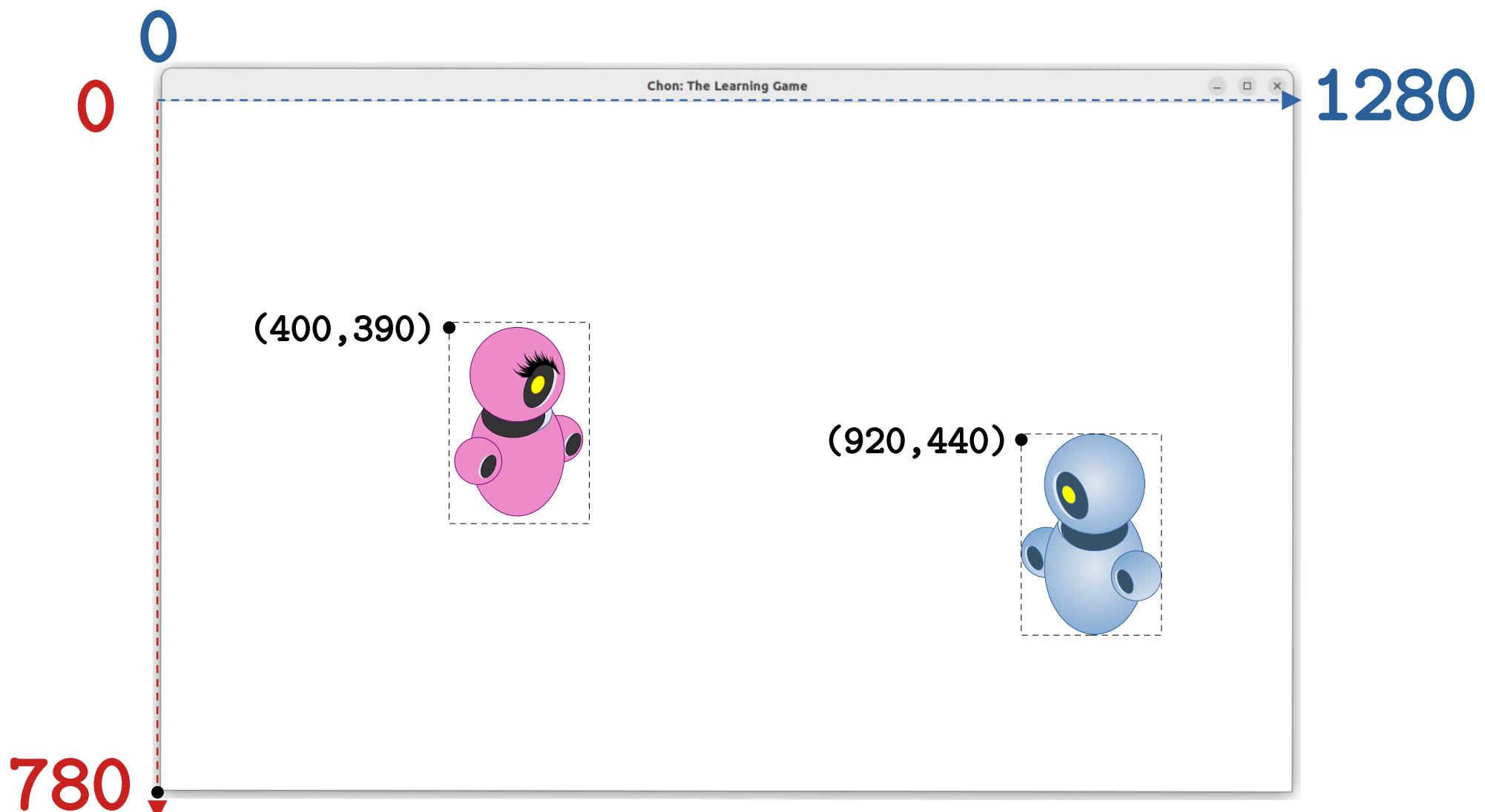
# Moving Another Object to the LEFT



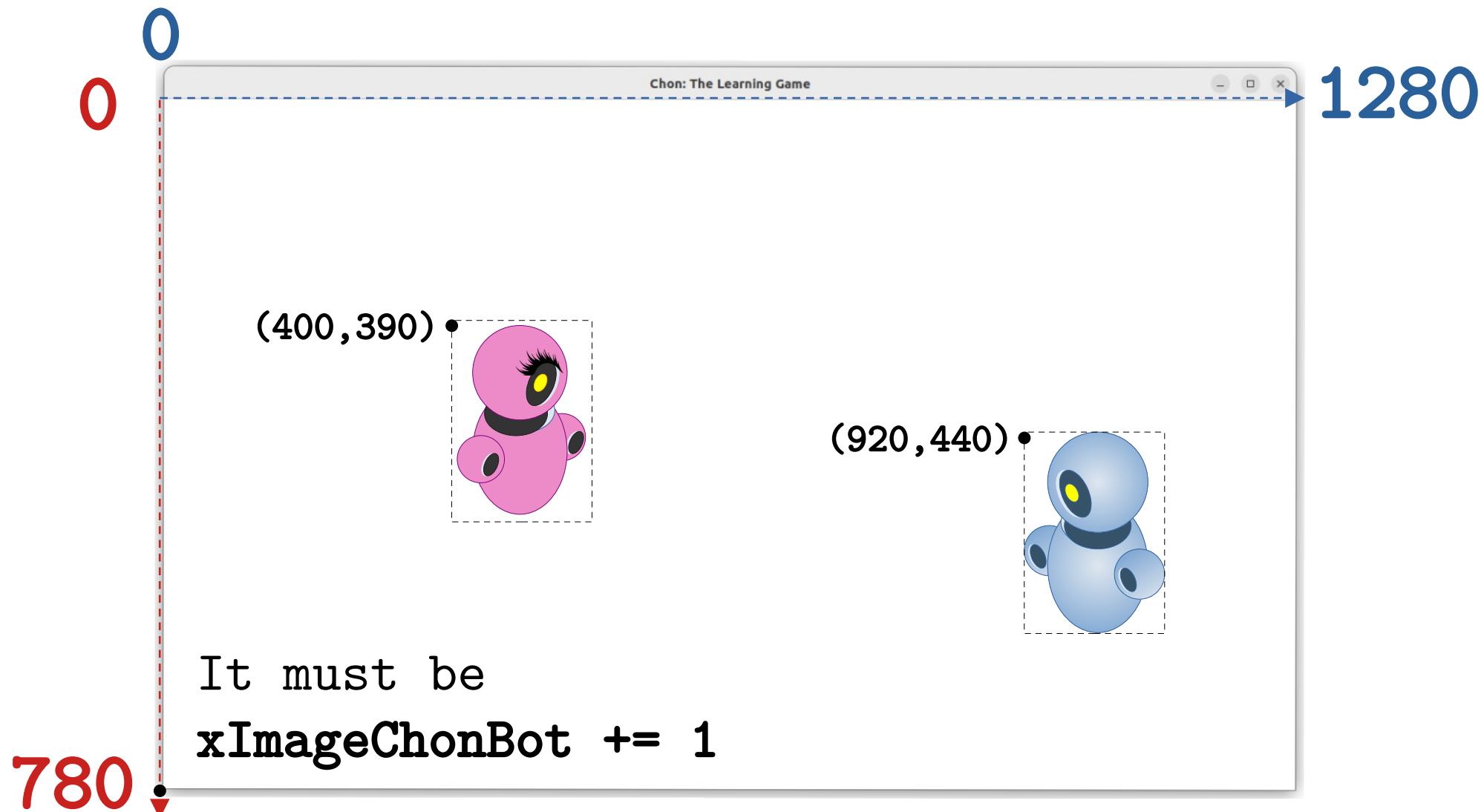
# Moving Another Object to the LEFT



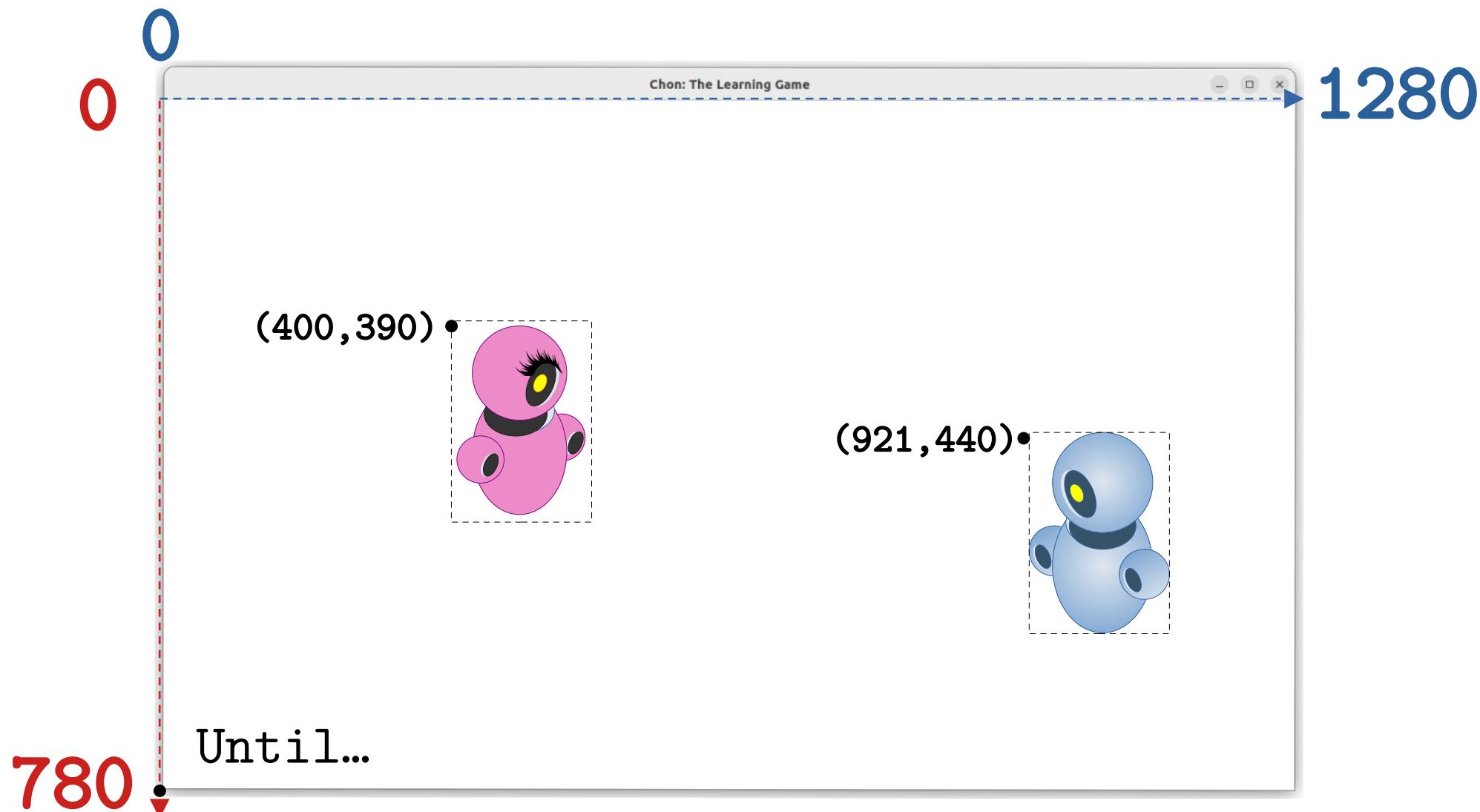
# Moving Another Object to the RIGHT



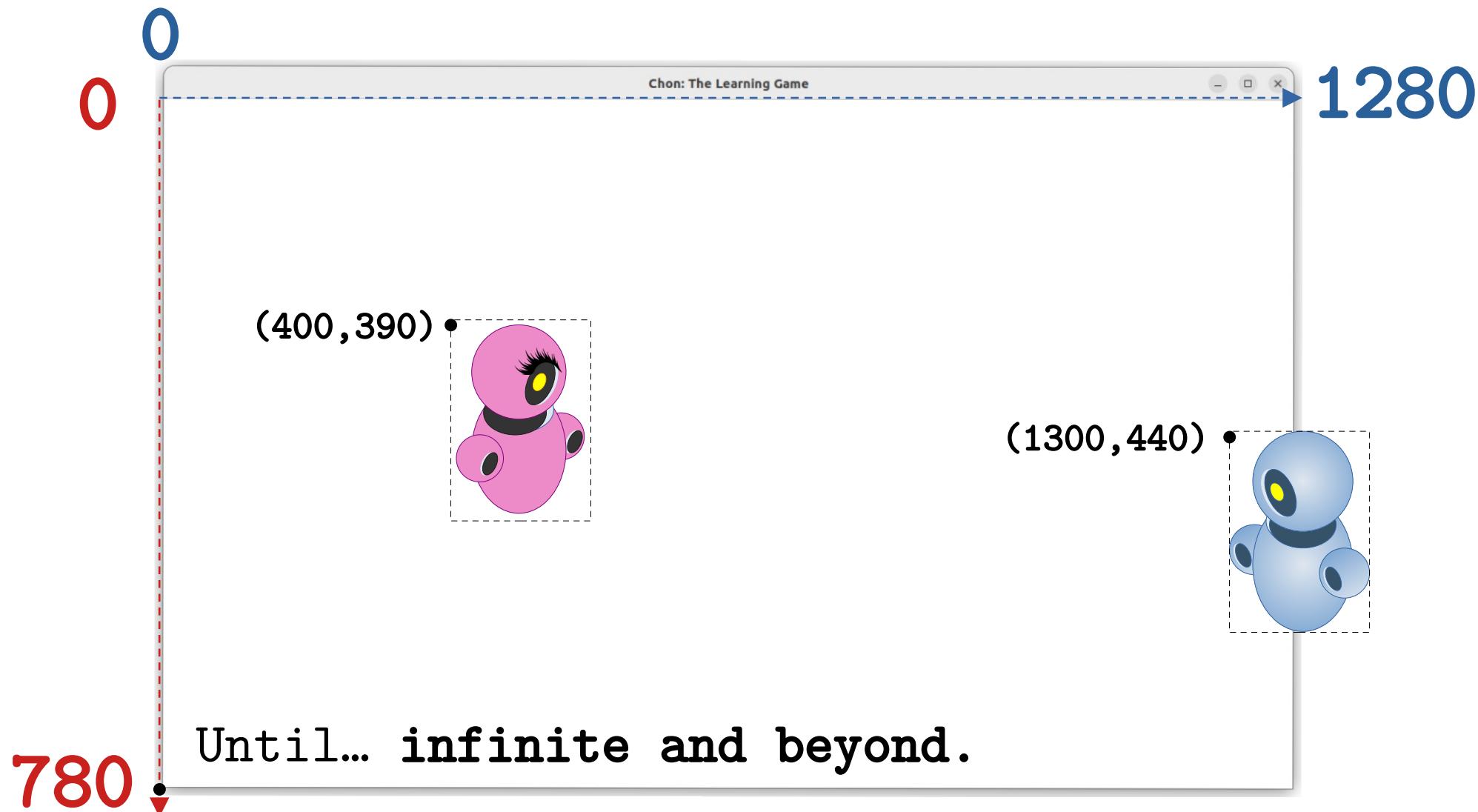
# Moving Another Object to the RIGHT



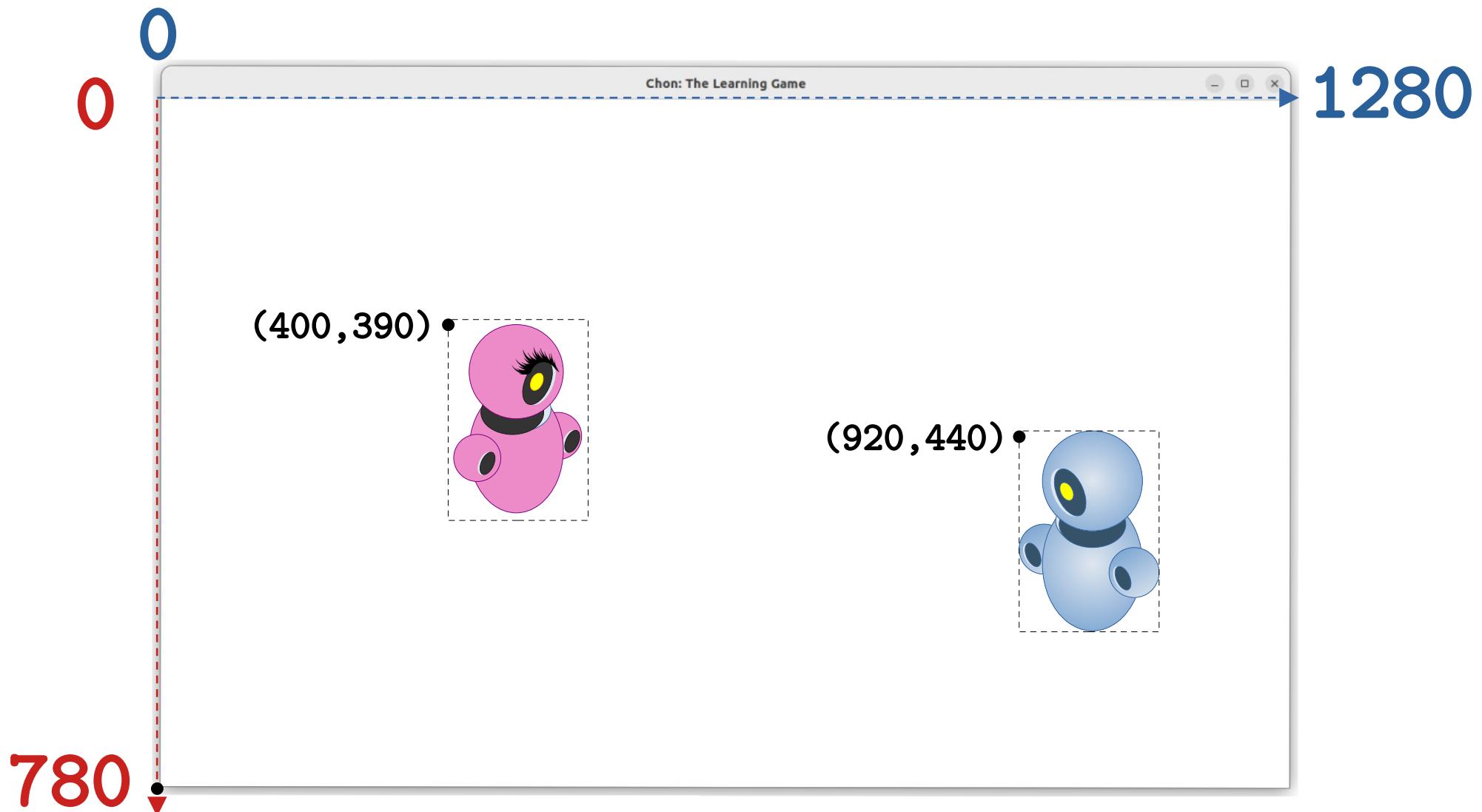
# Moving Another Object to the RIGHT



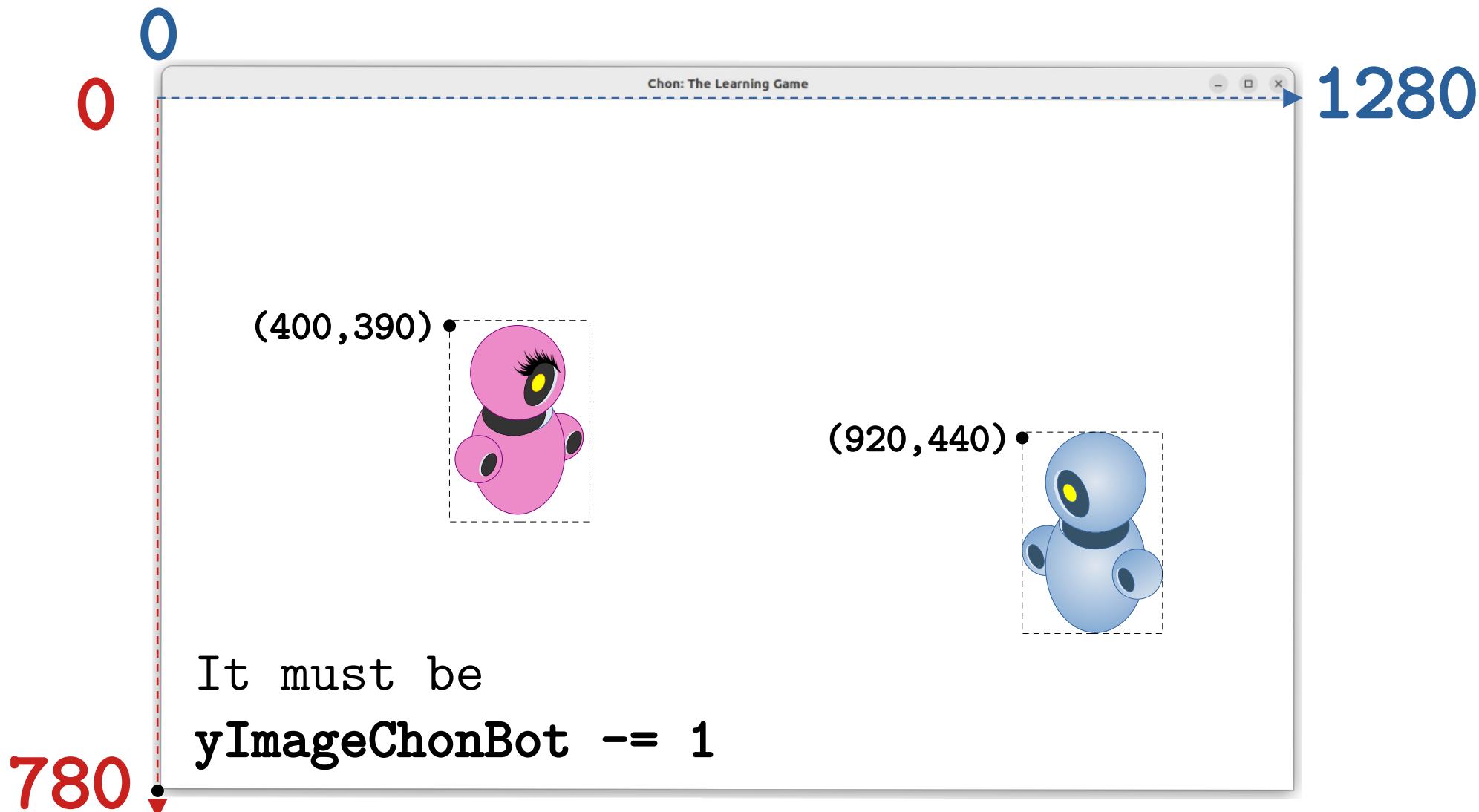
# Moving Another Object to the RIGHT



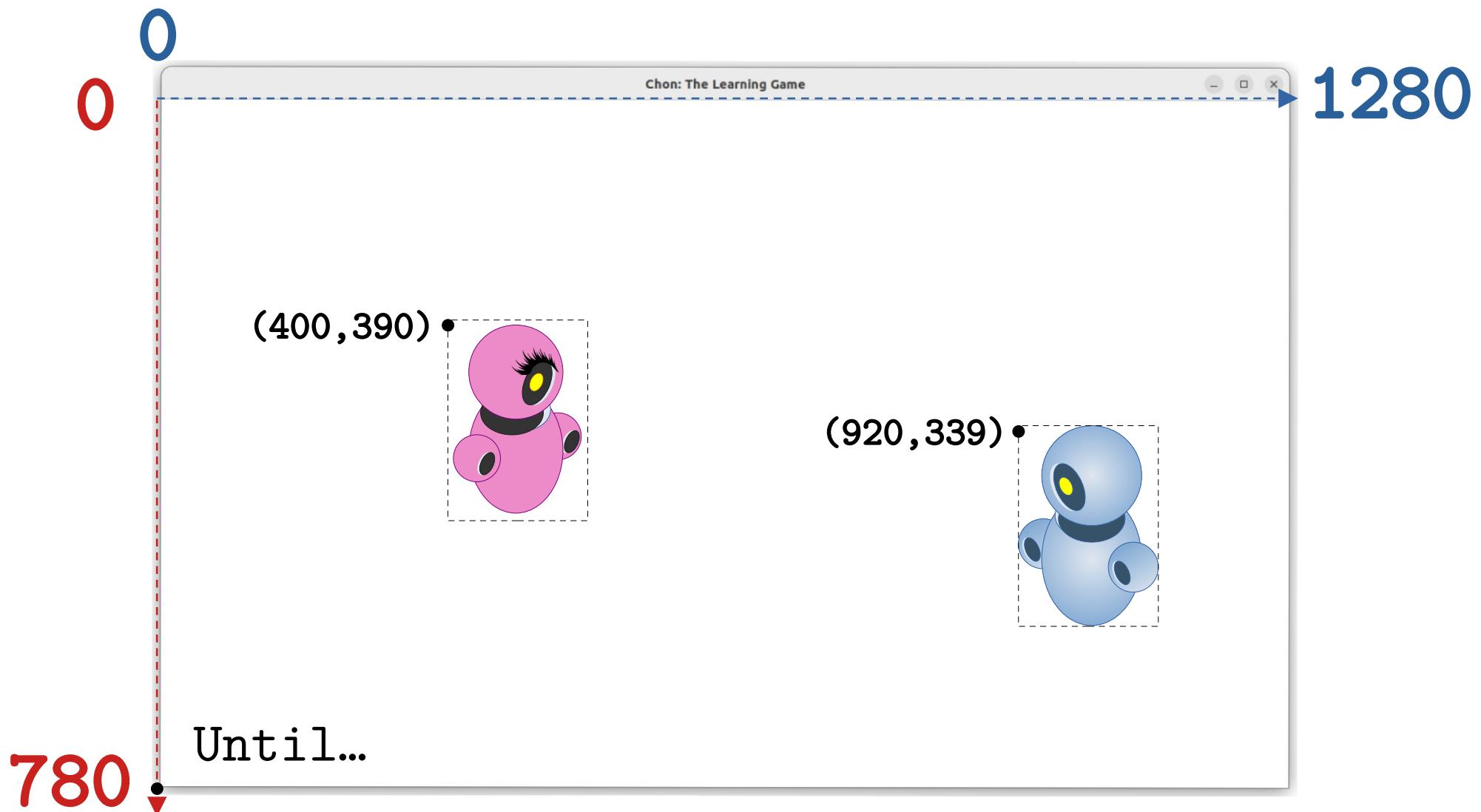
# Moving UP Another Object



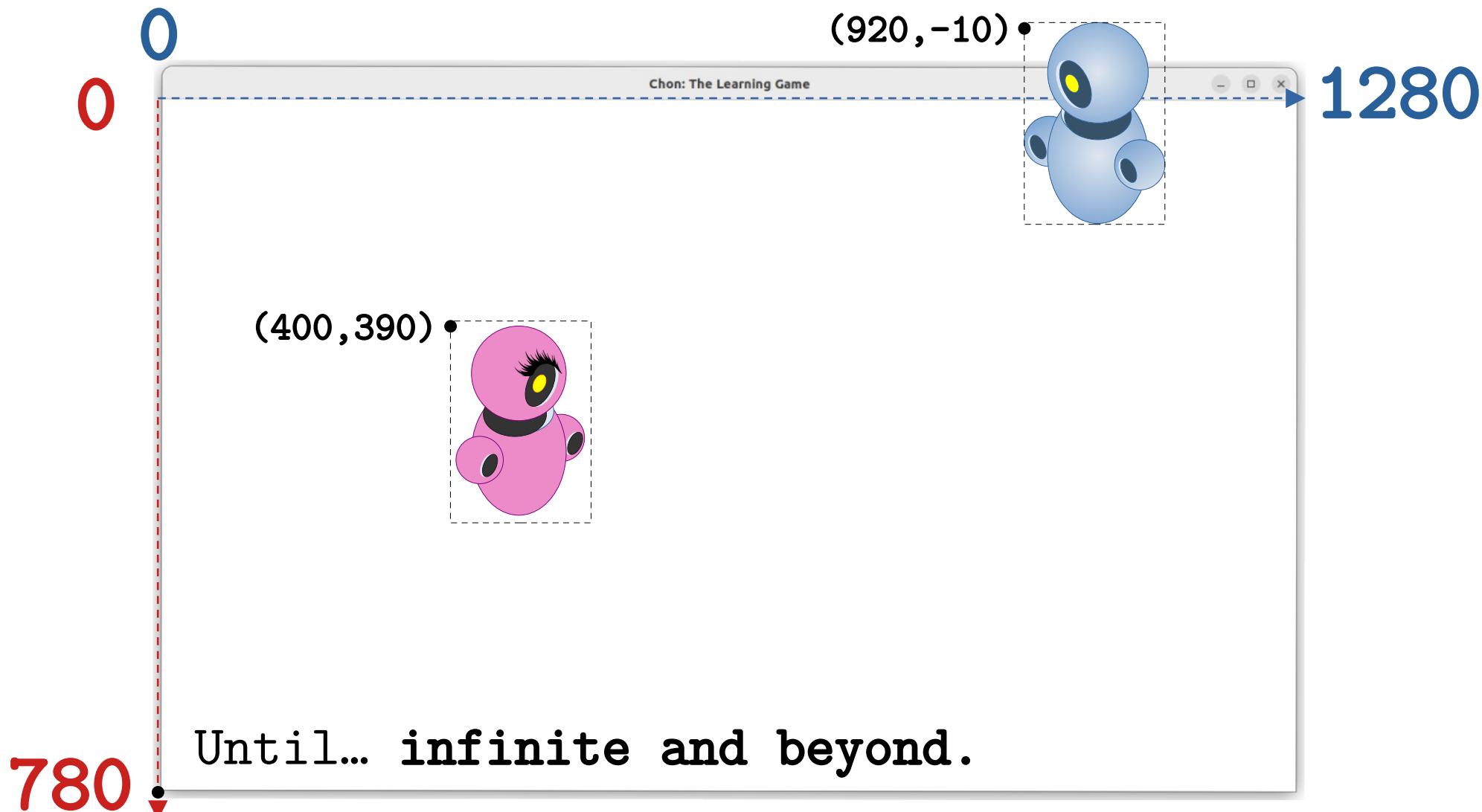
# Moving UP Another Object



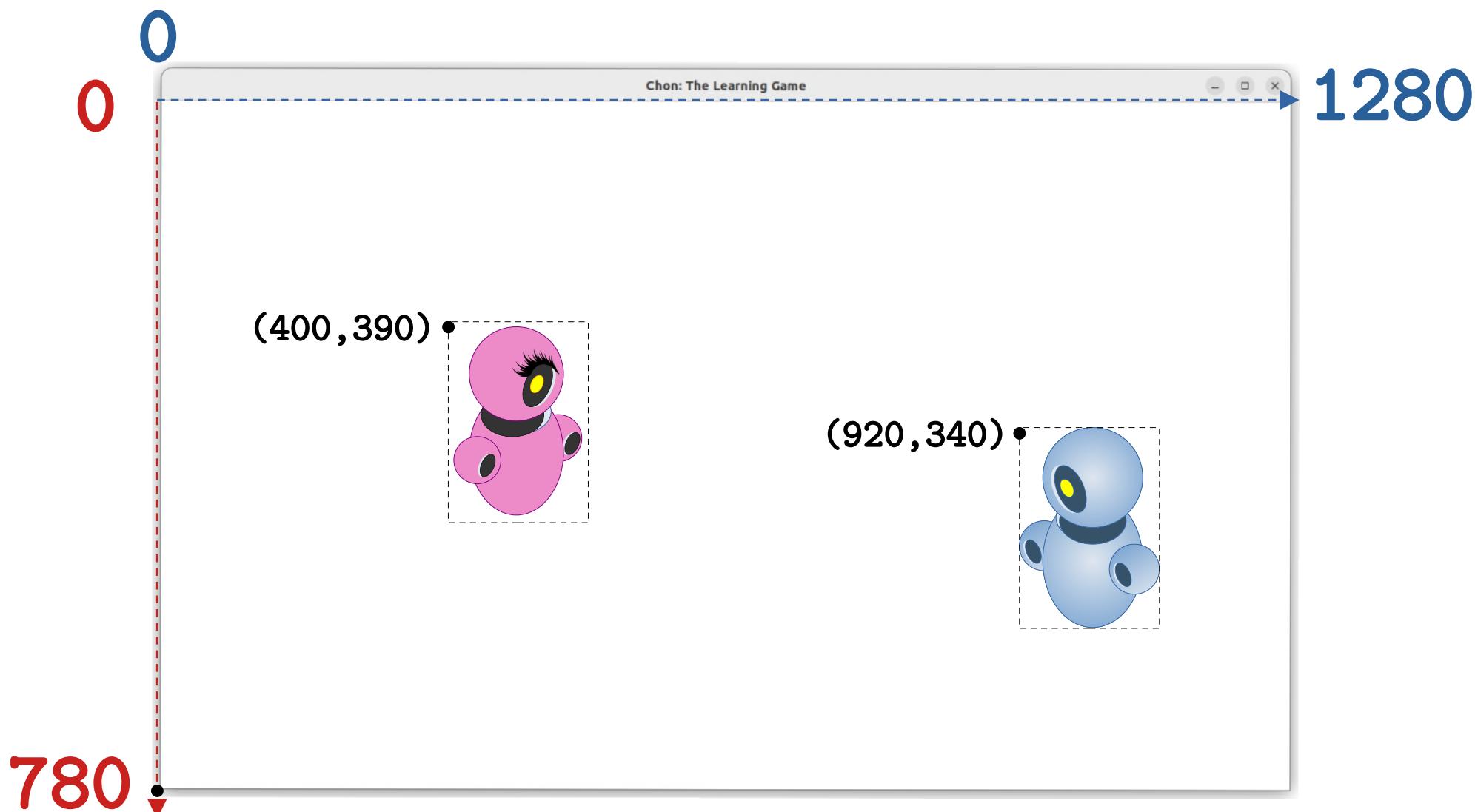
# Moving UP Another Object



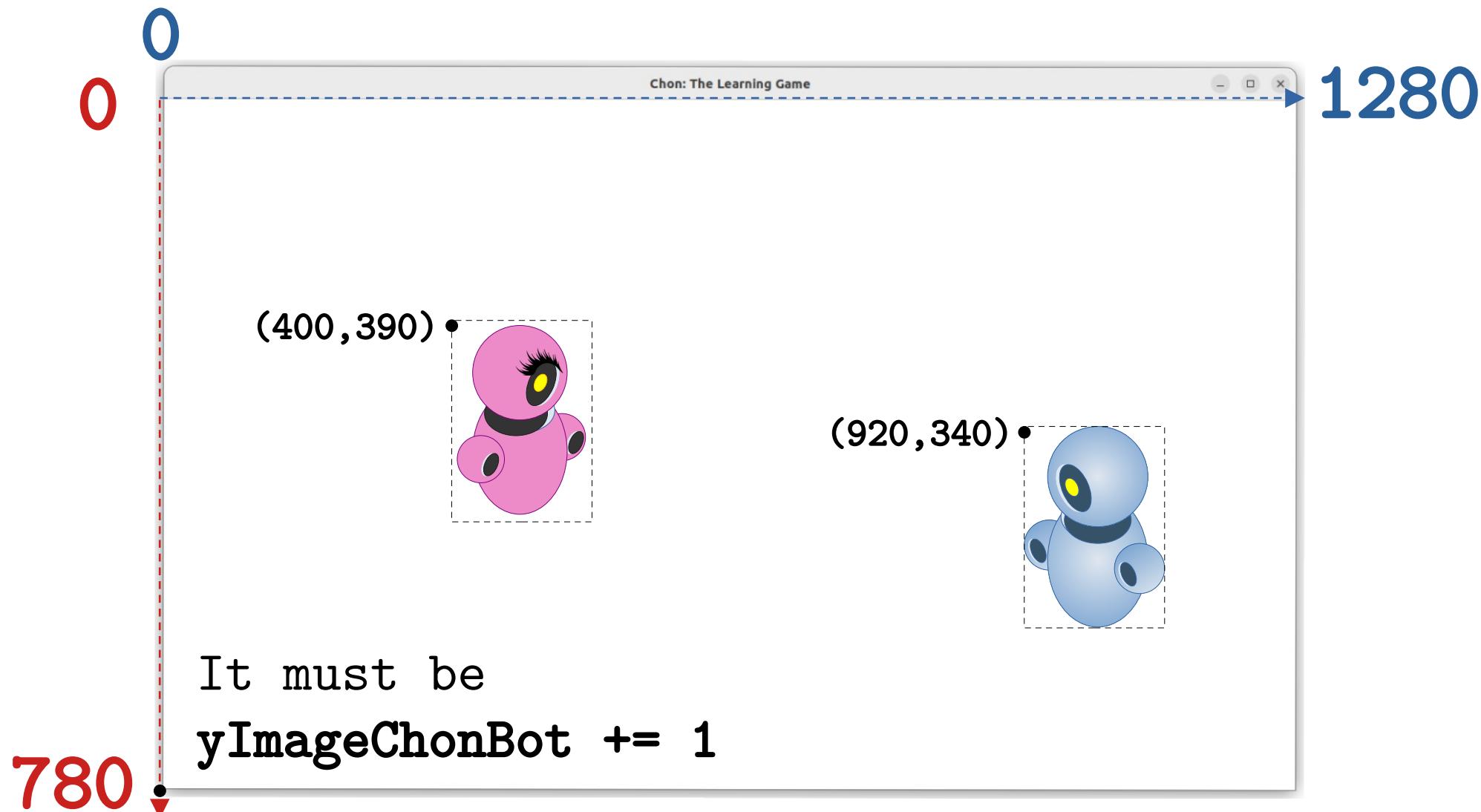
# Moving UP Another Object



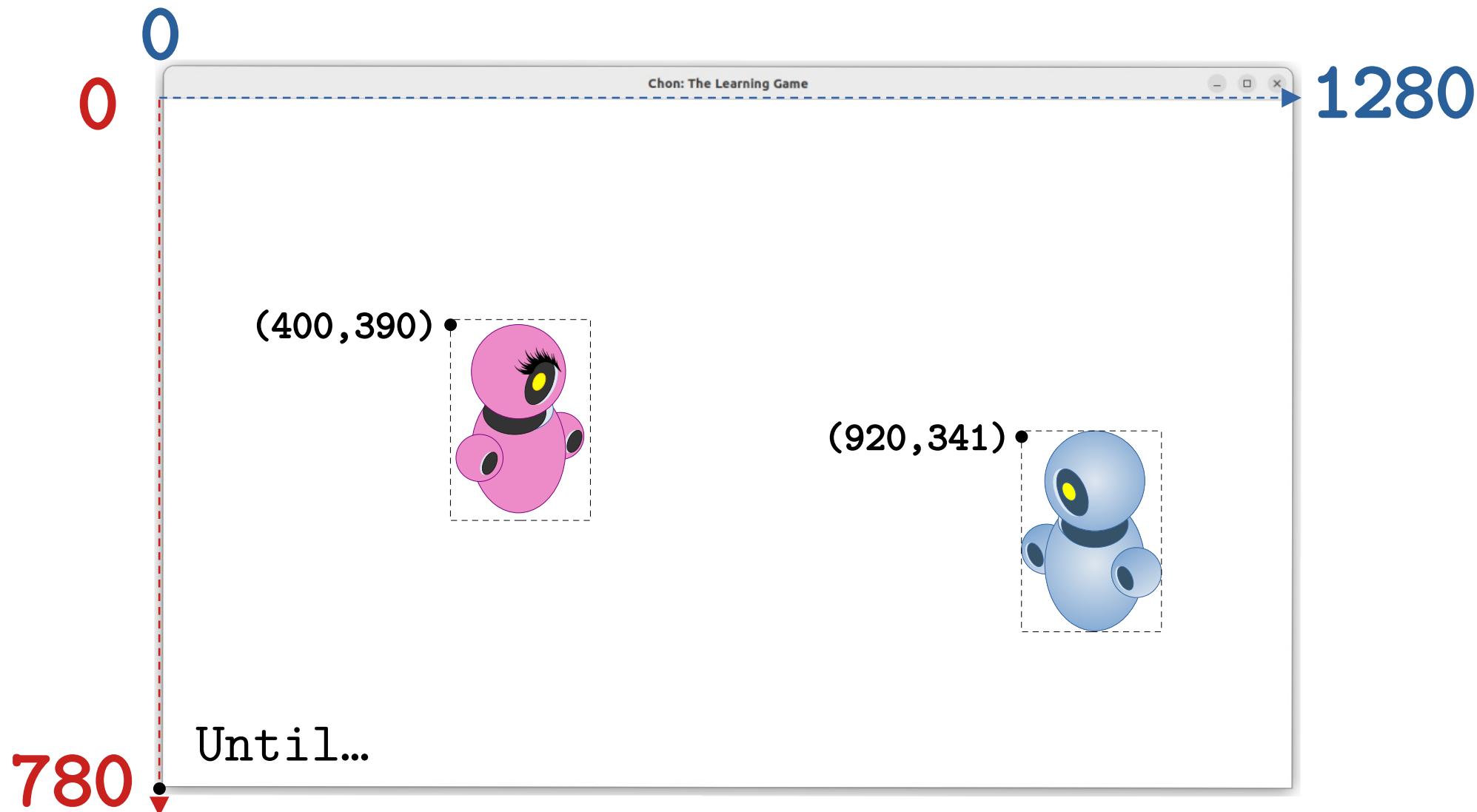
# Moving DOWN Another Object



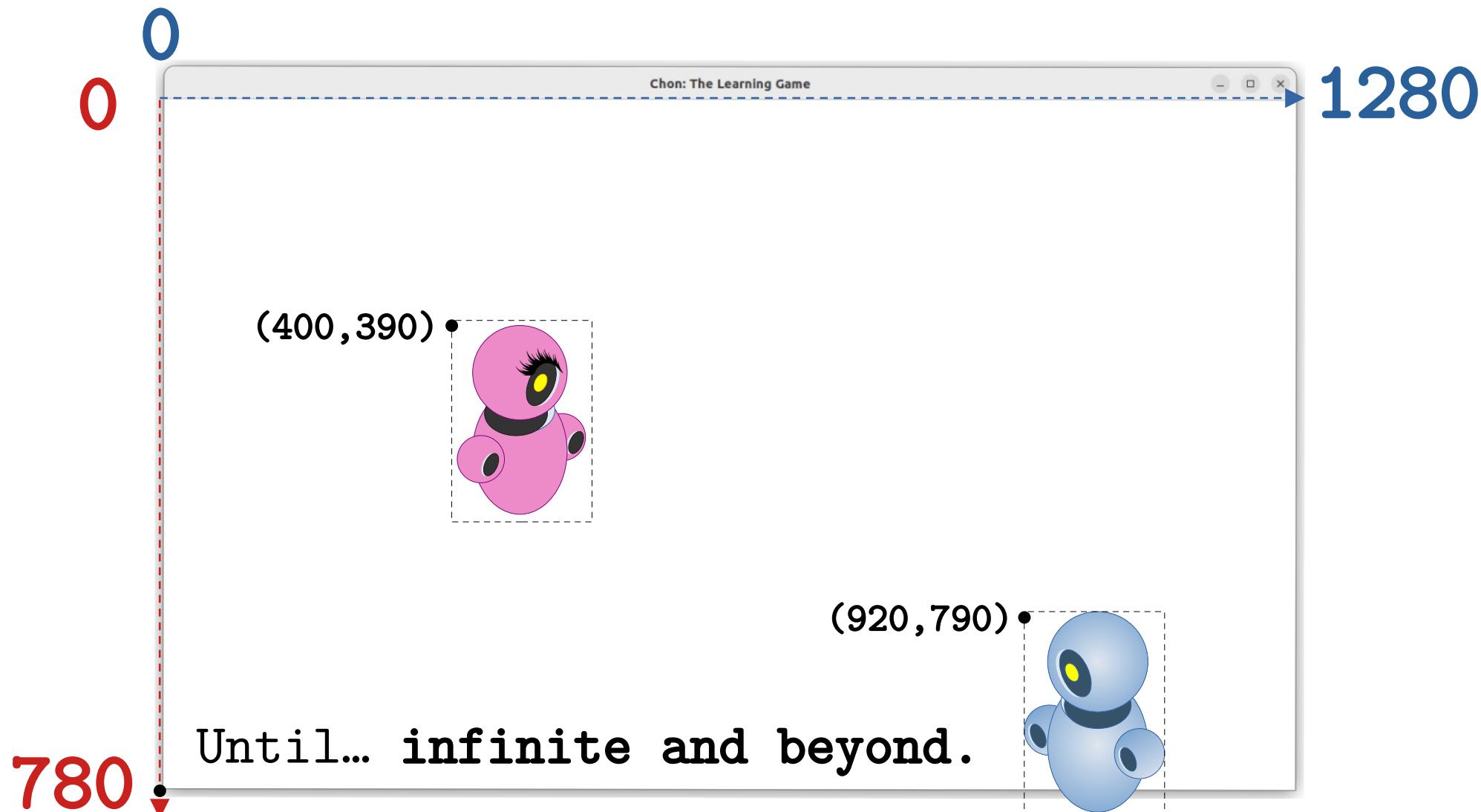
# Moving DOWN Another Object



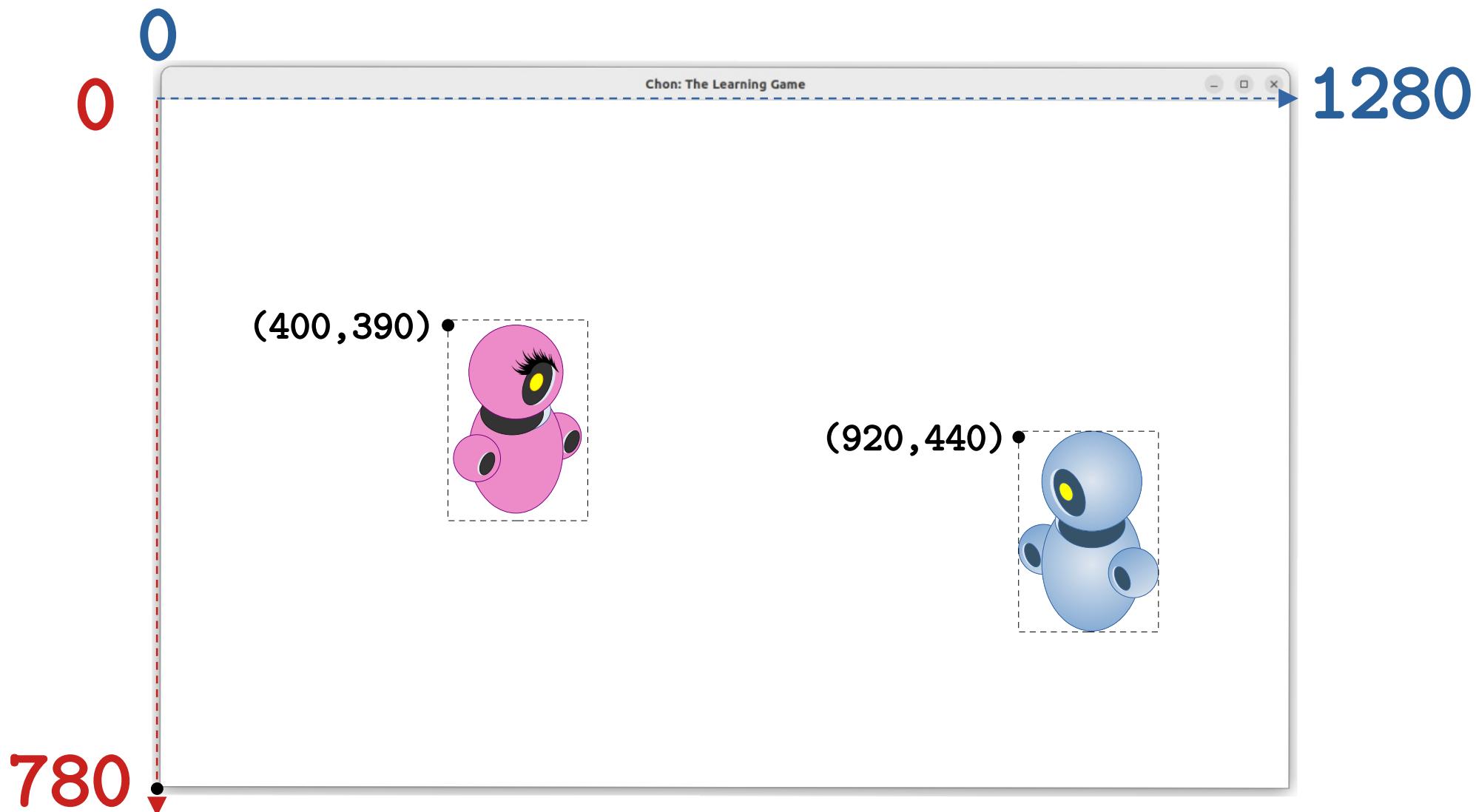
# Moving DOWN Another Object



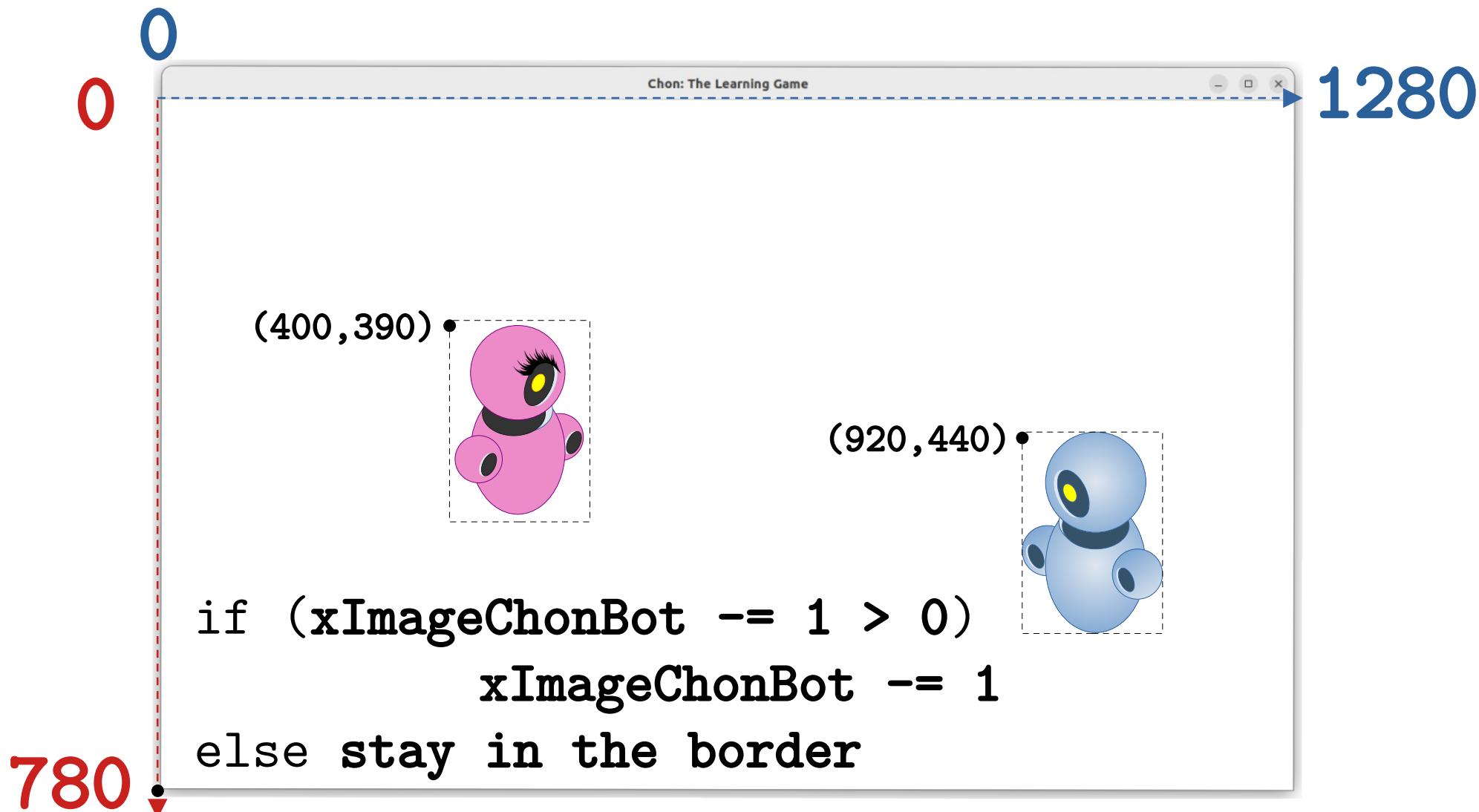
# Moving DOWN Another Object



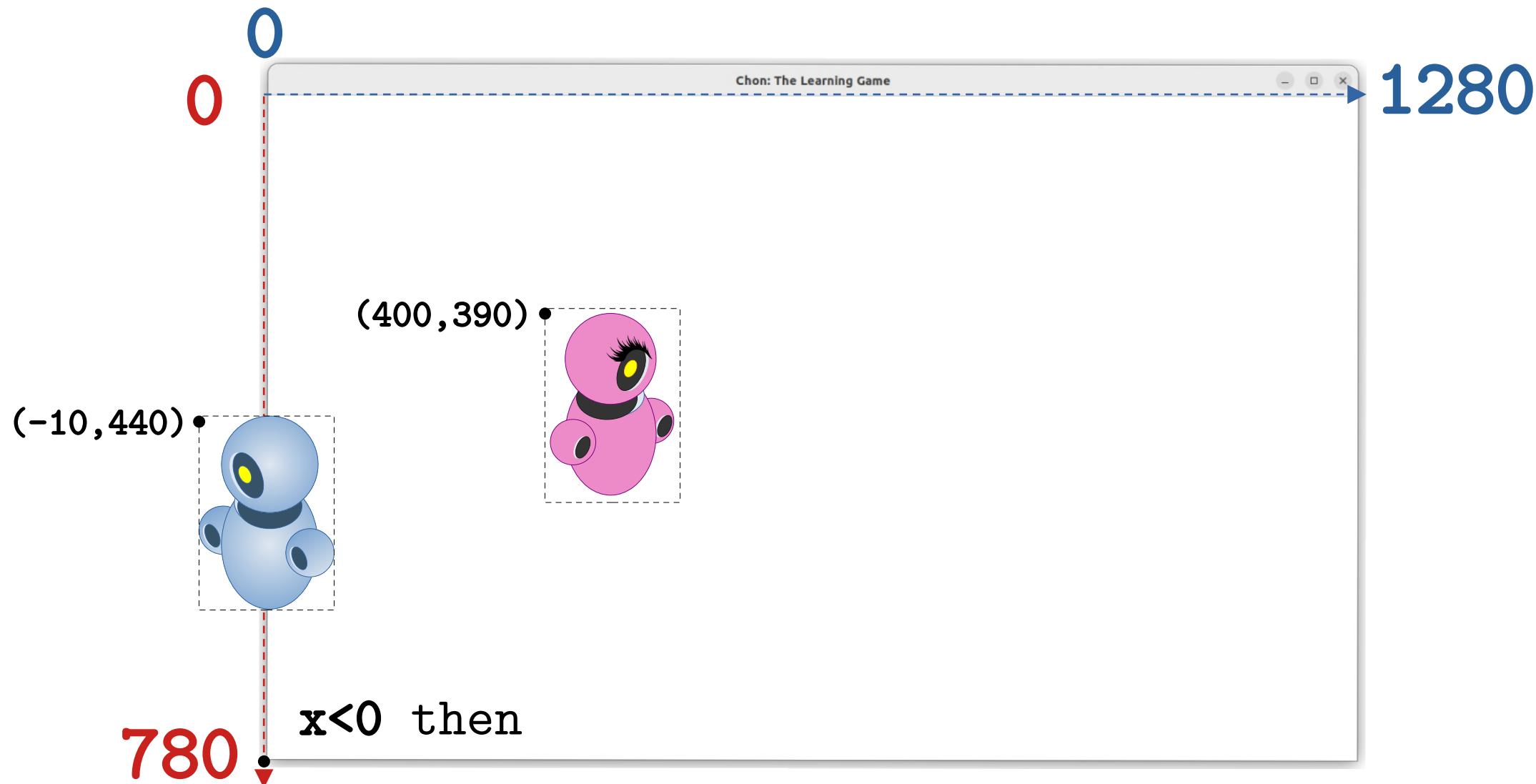
# Moving Until the Border



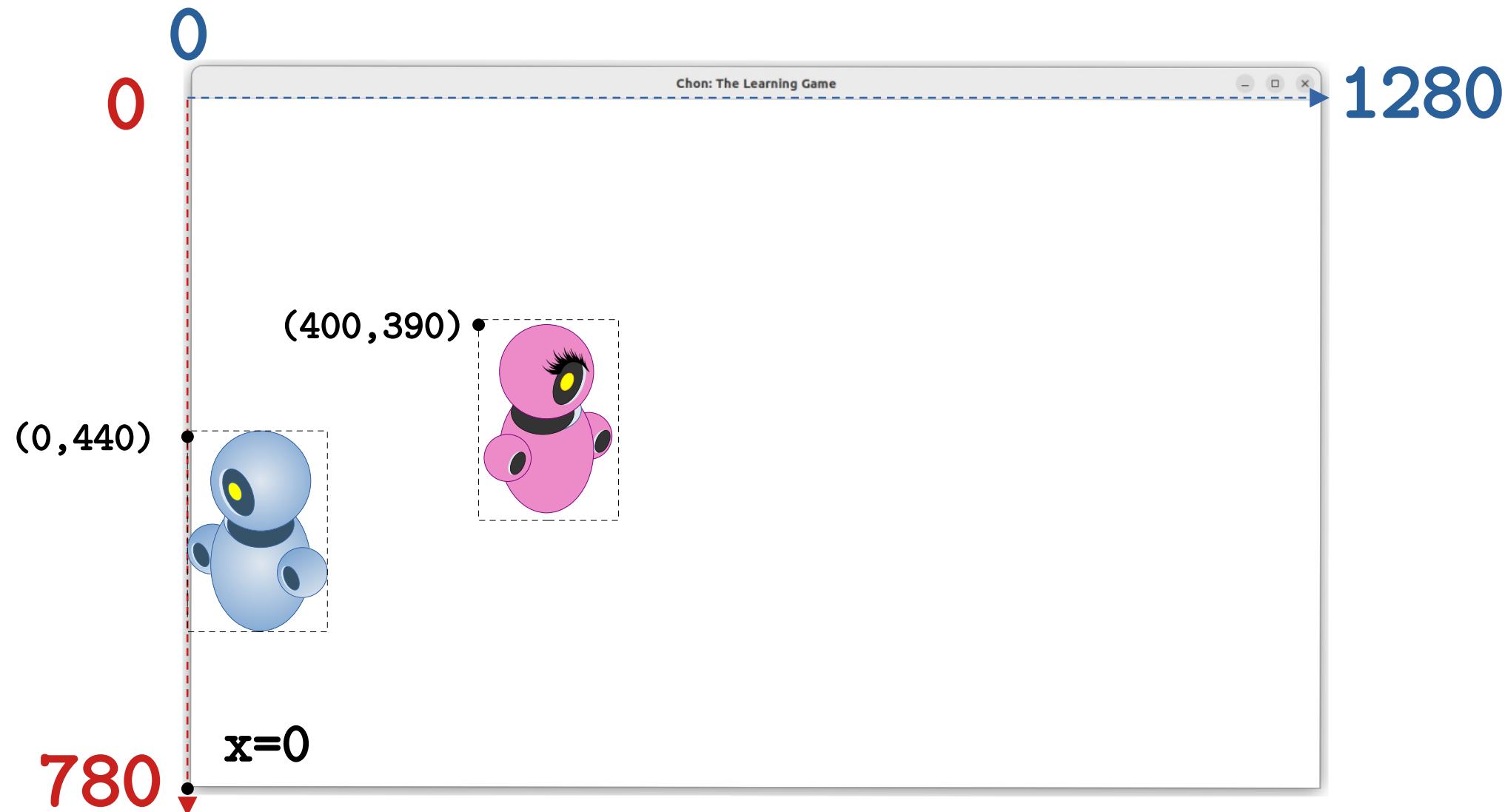
# Moving Until the Border



# Moving Until the Border



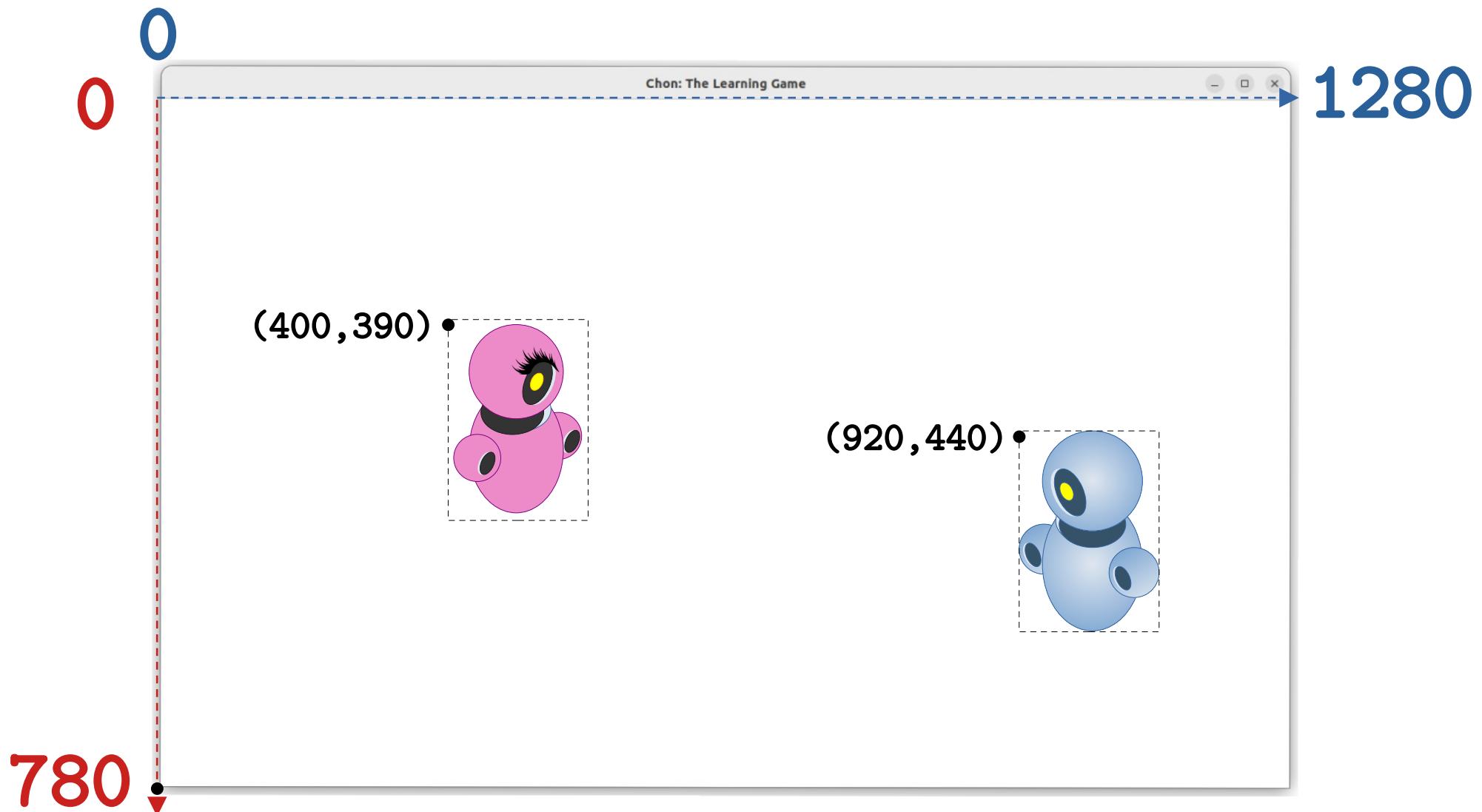
# Moving Until the Border



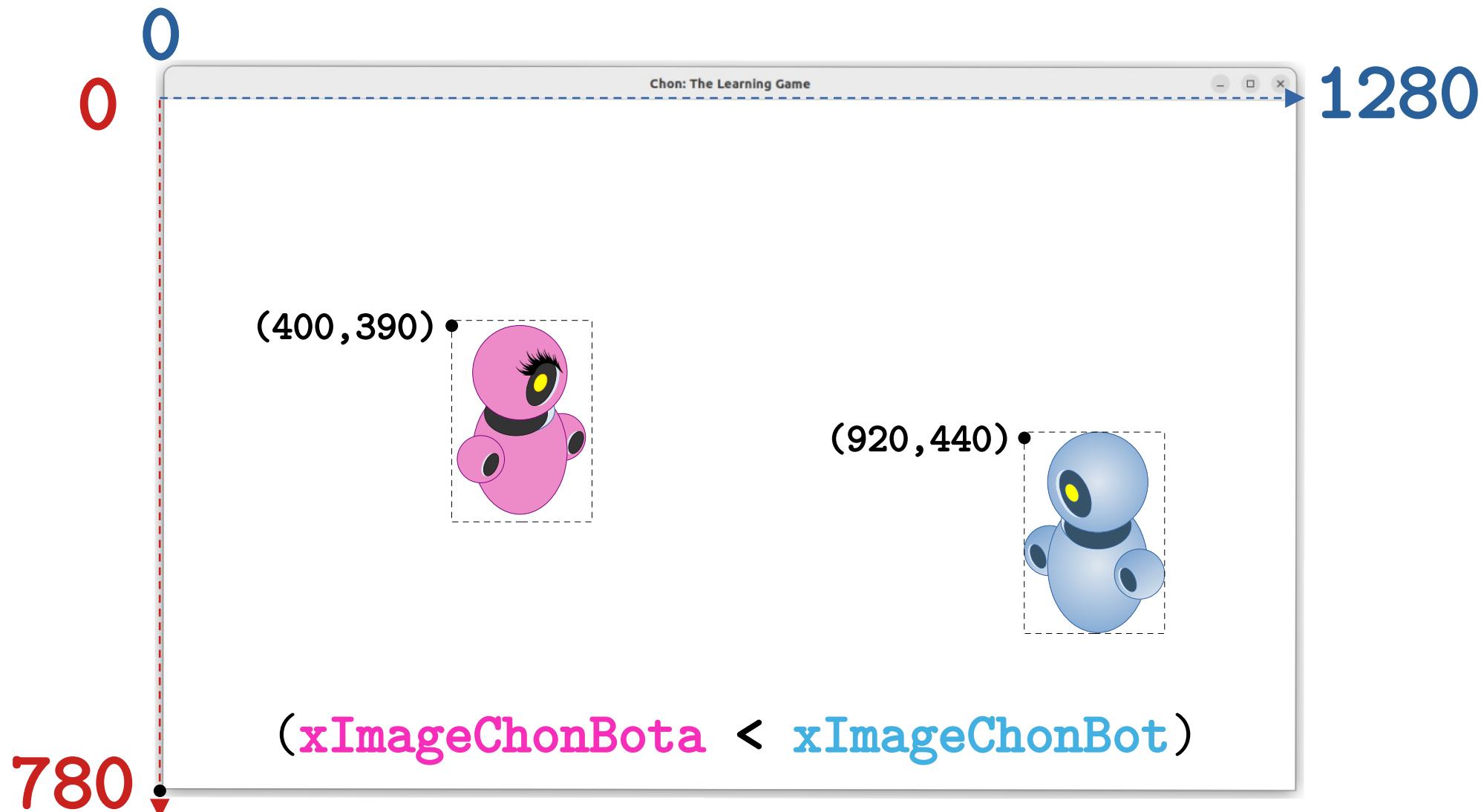
# Moving Until the Border

The same mechanics  
applies to the  
**RIGHT, UP and DOWN.**

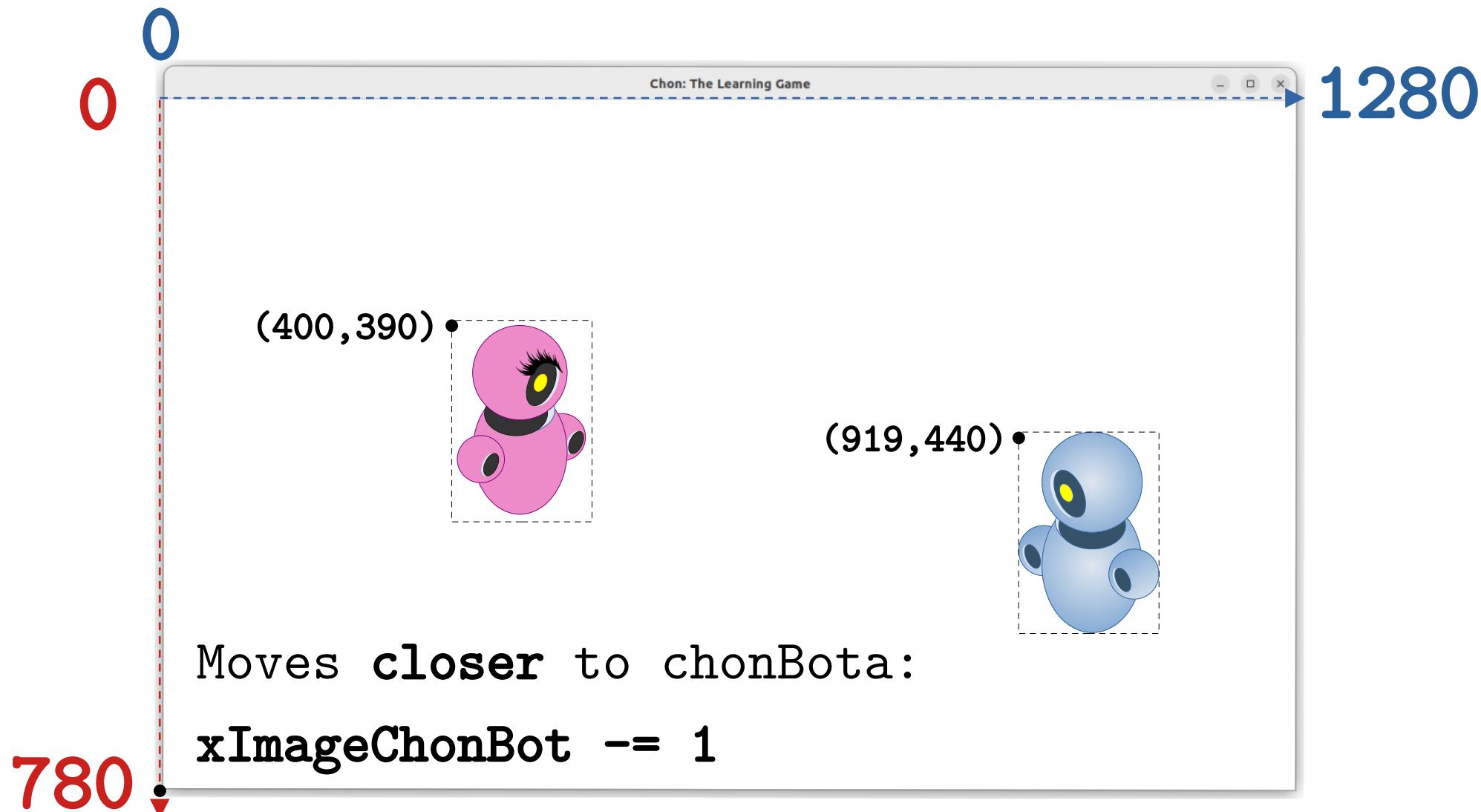
# Chasing the Player



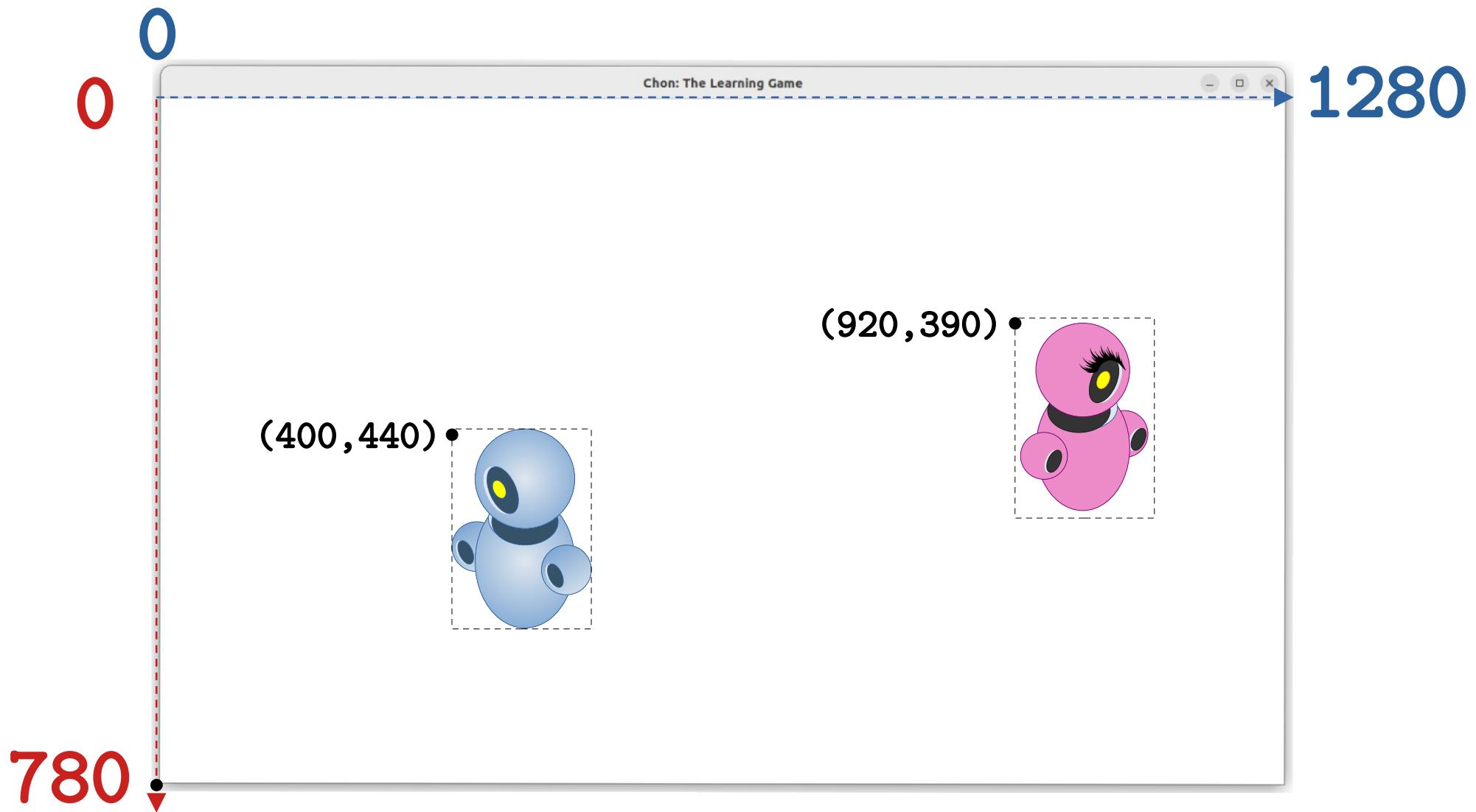
# Chasing the Player



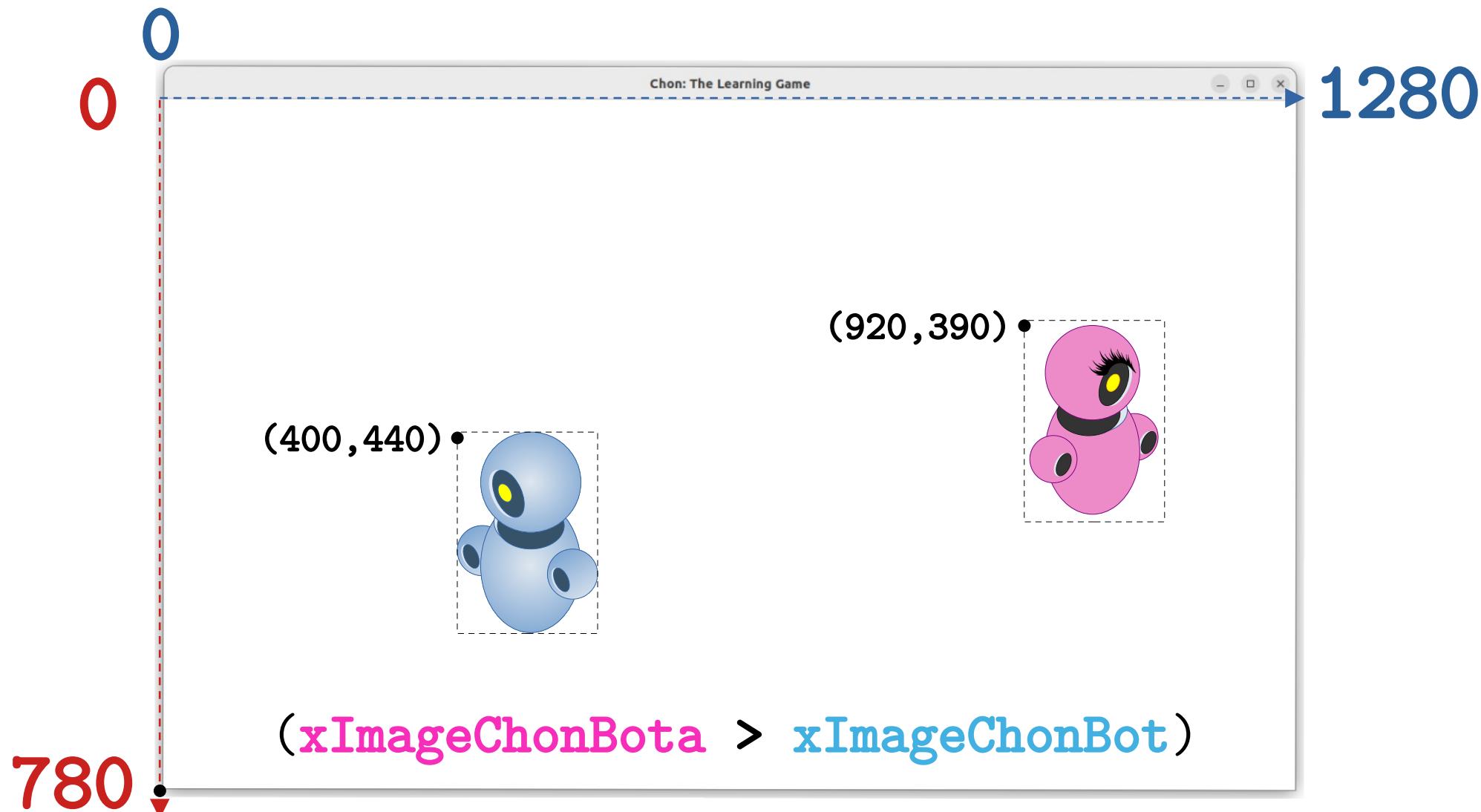
# Chasing the Player



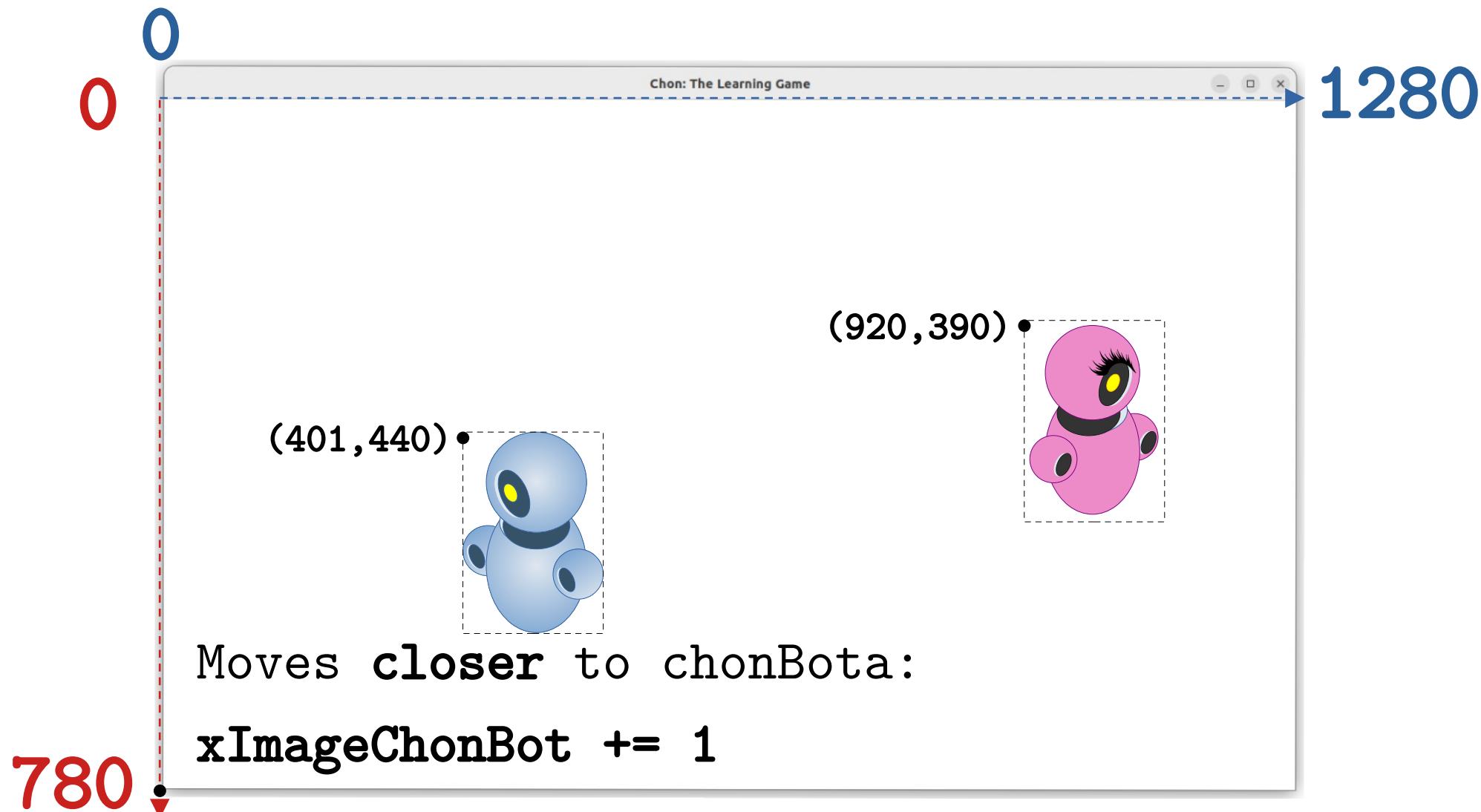
# Chasing the Player



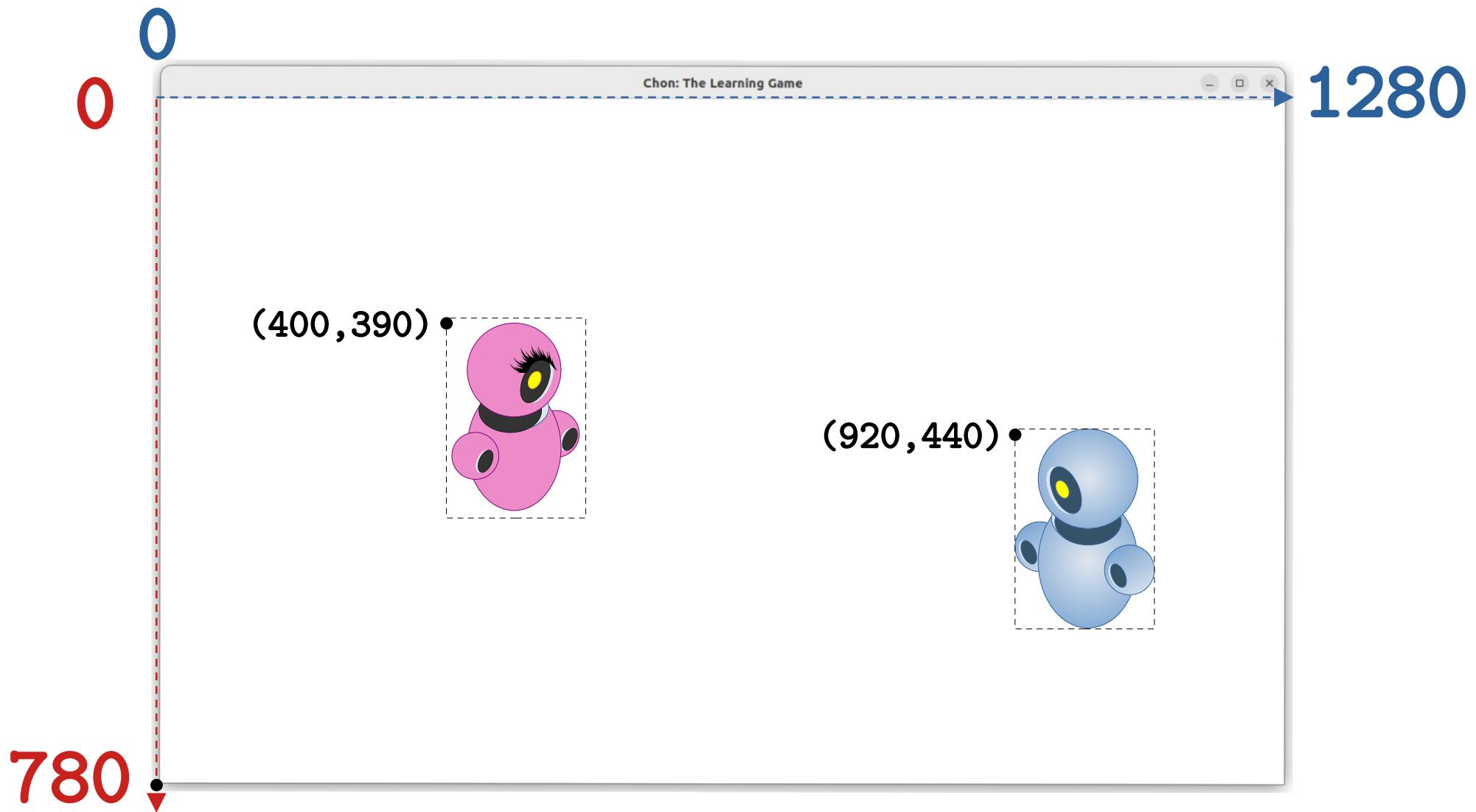
# Chasing the Player



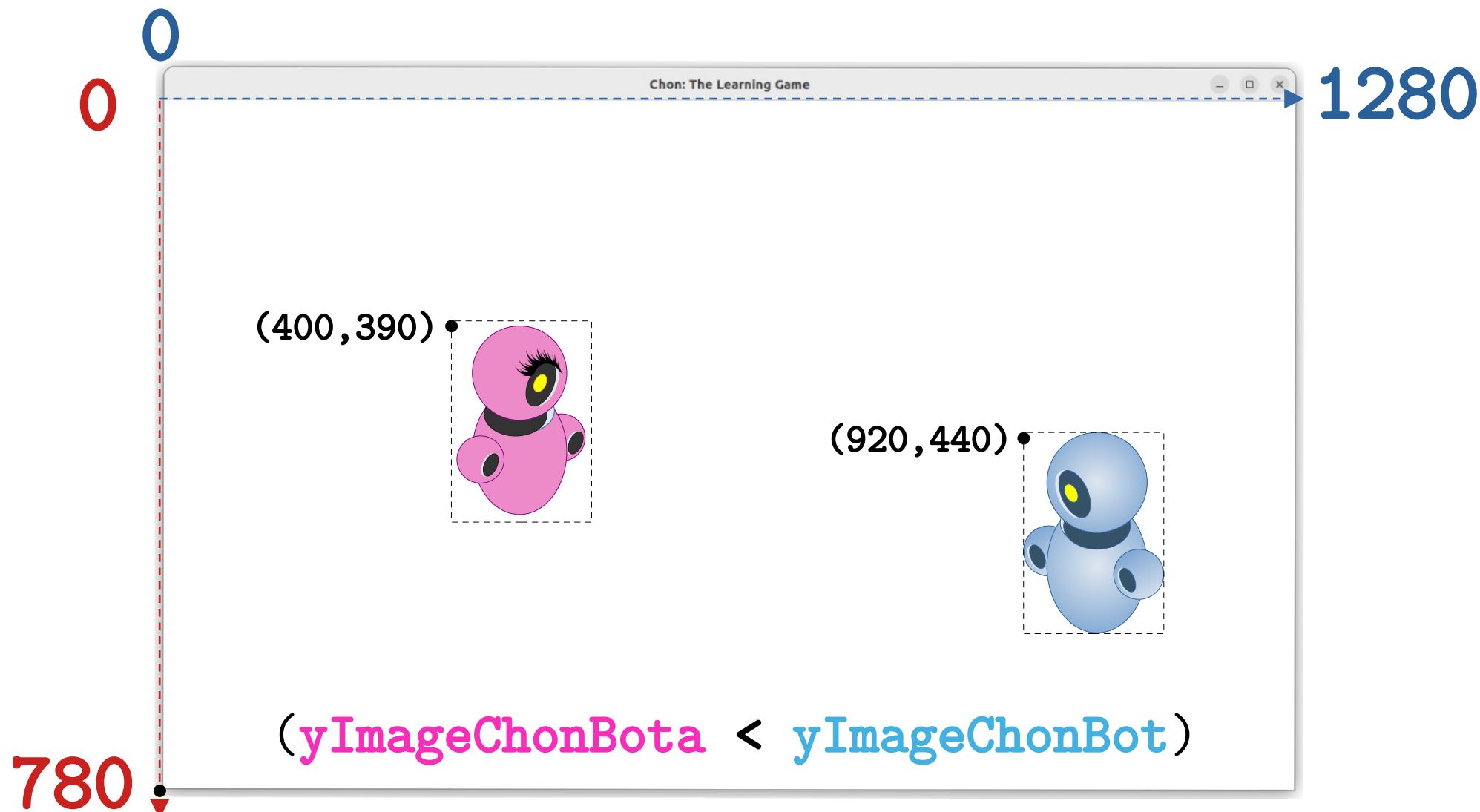
# Chasing the Player



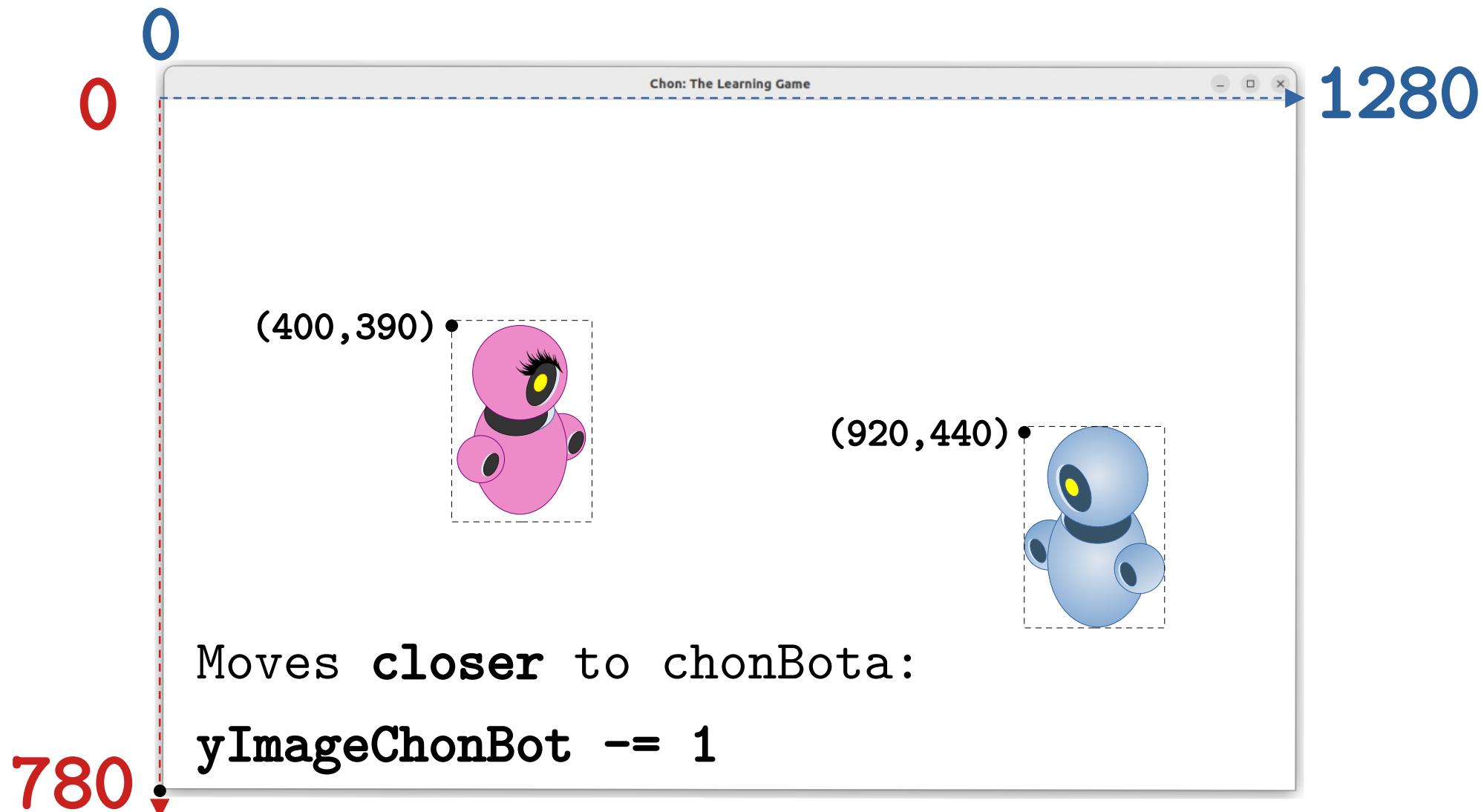
# Chasing the Player



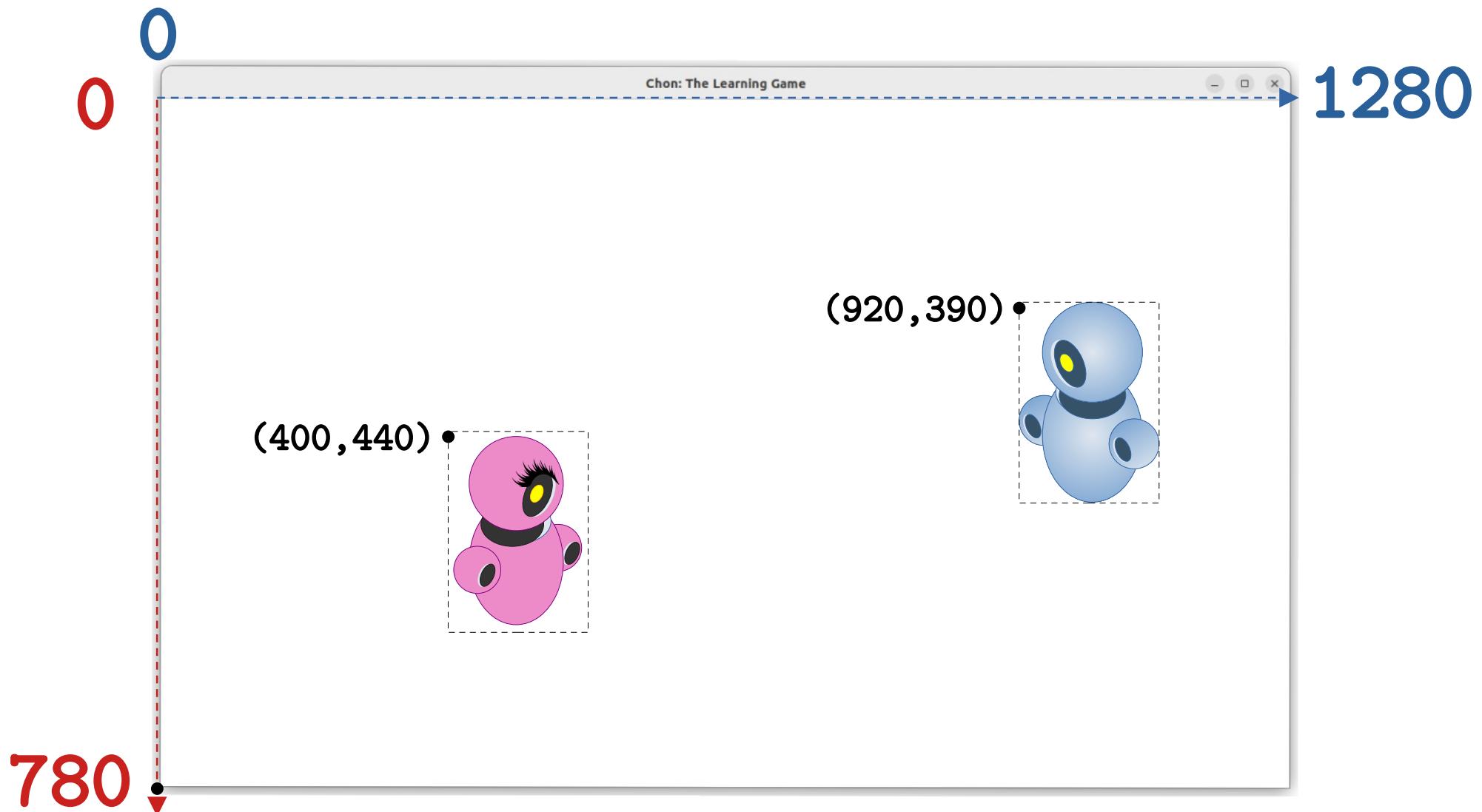
# Chasing the Player



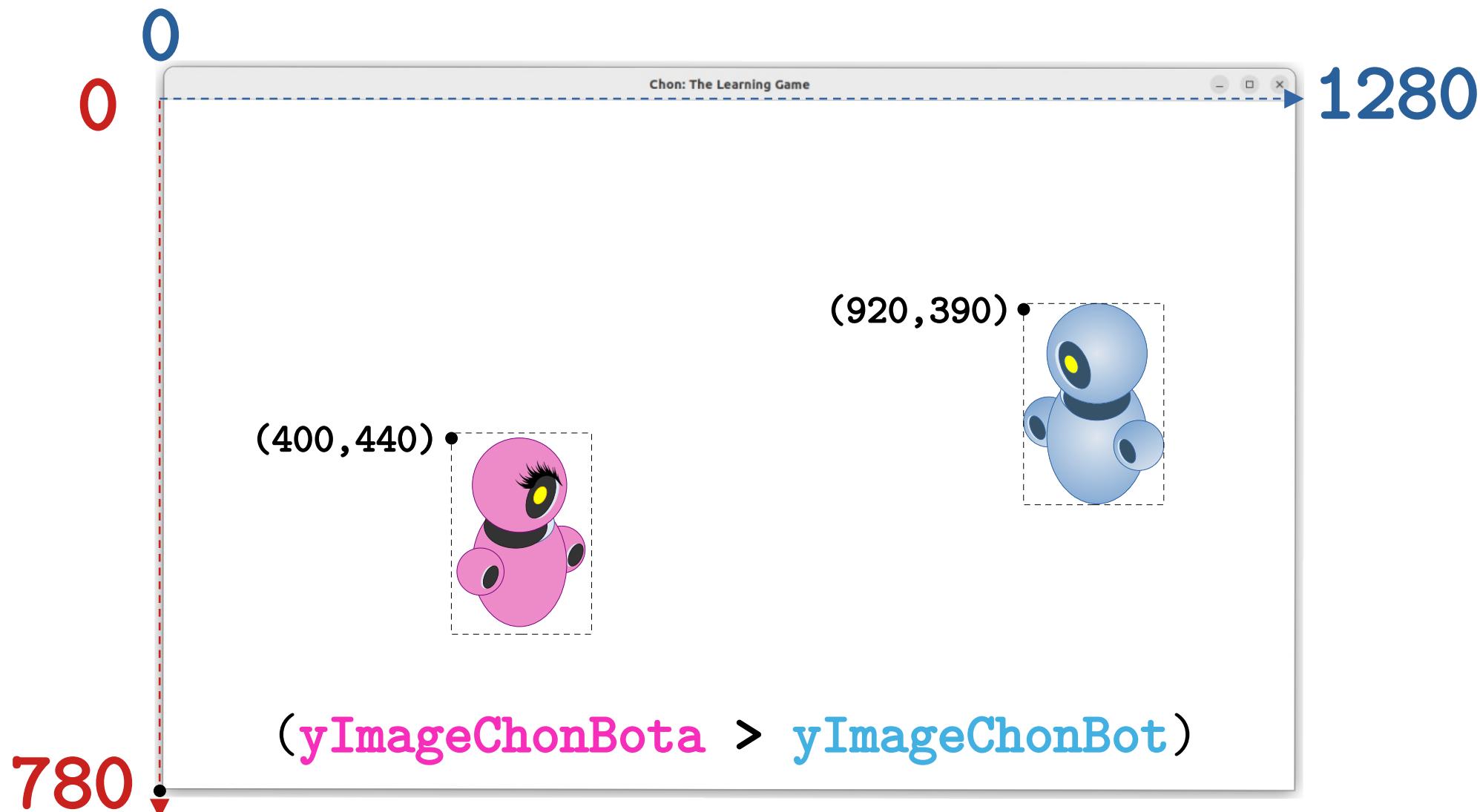
# Chasing the Player



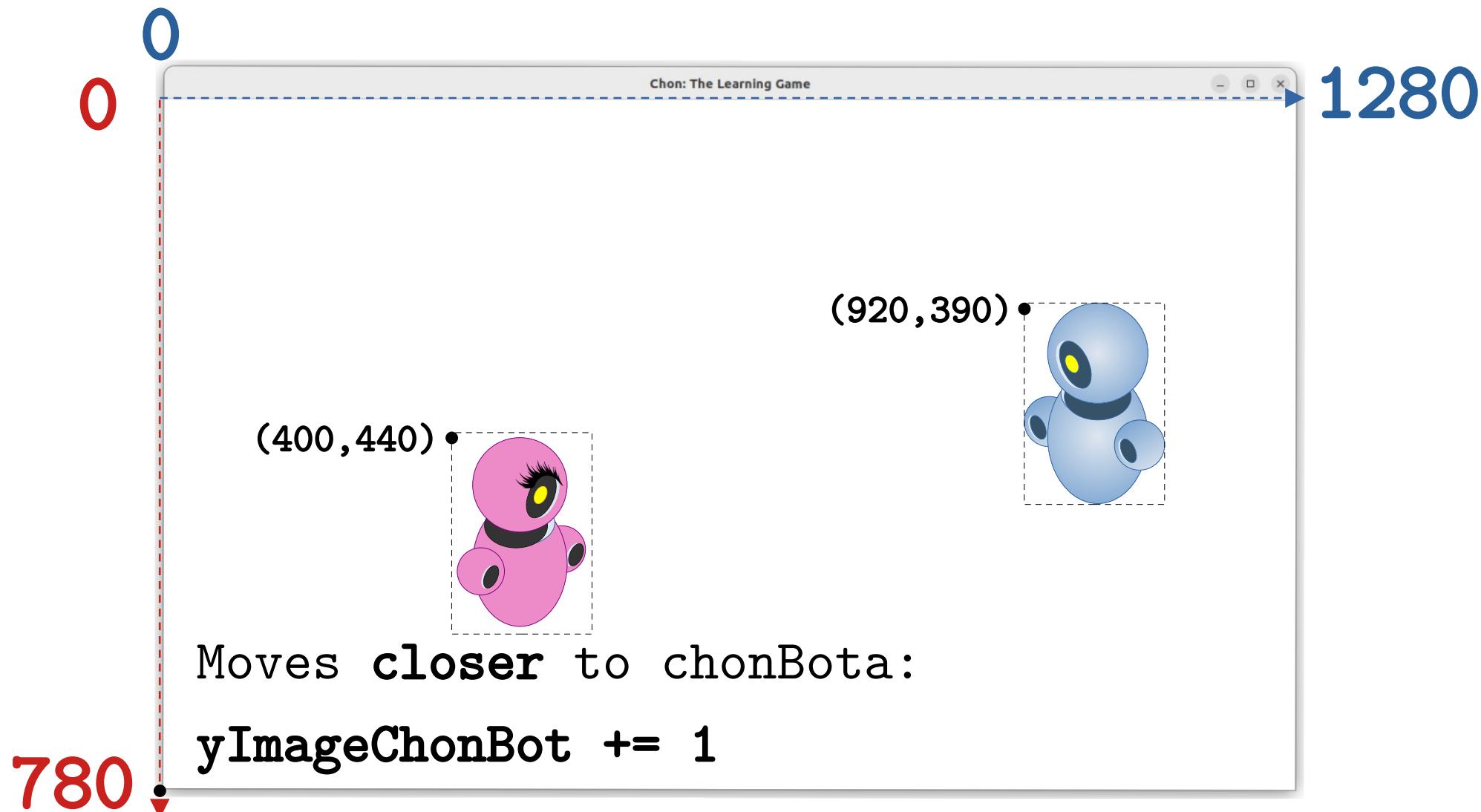
# Chasing the Player



# Chasing the Player



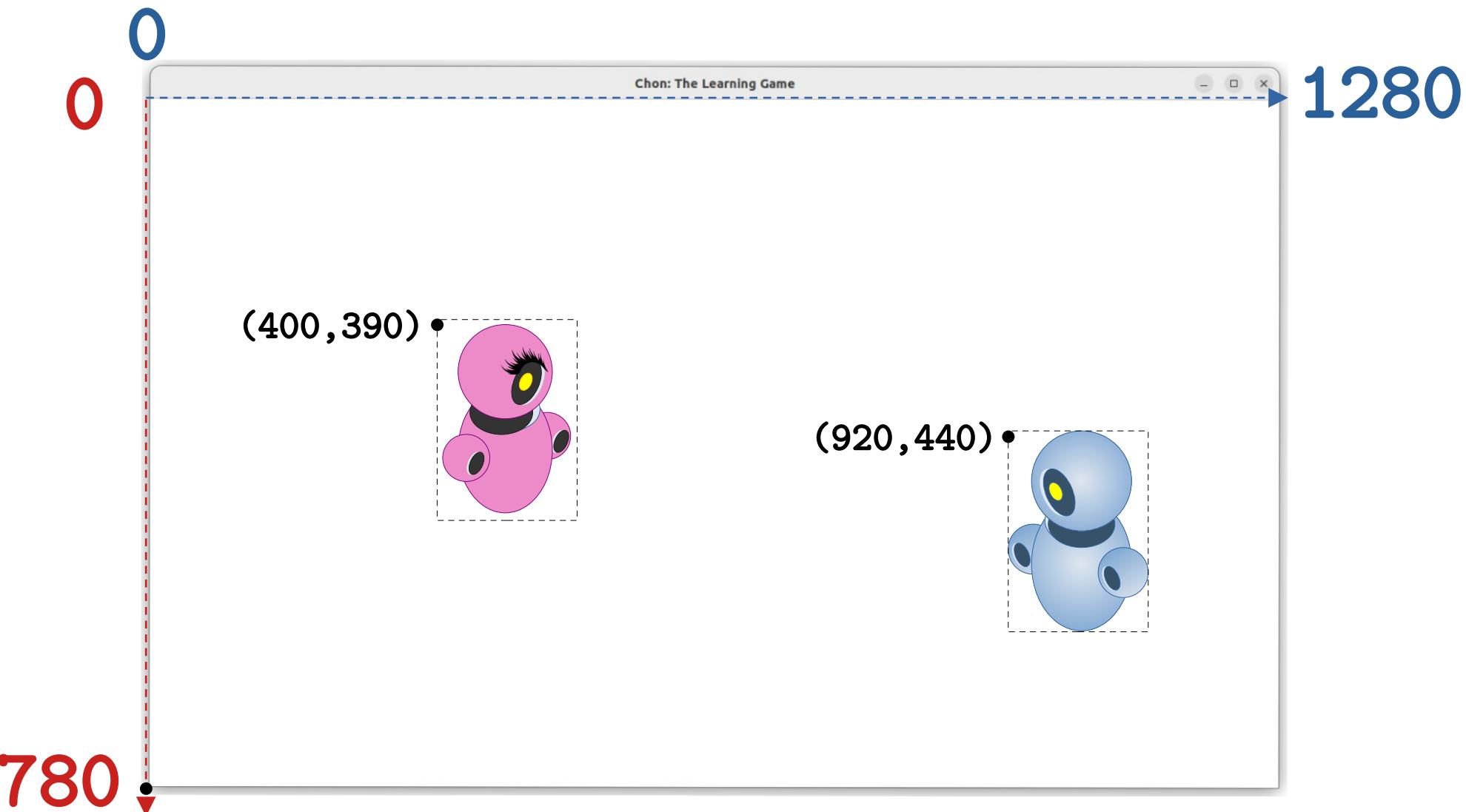
# Chasing the Player



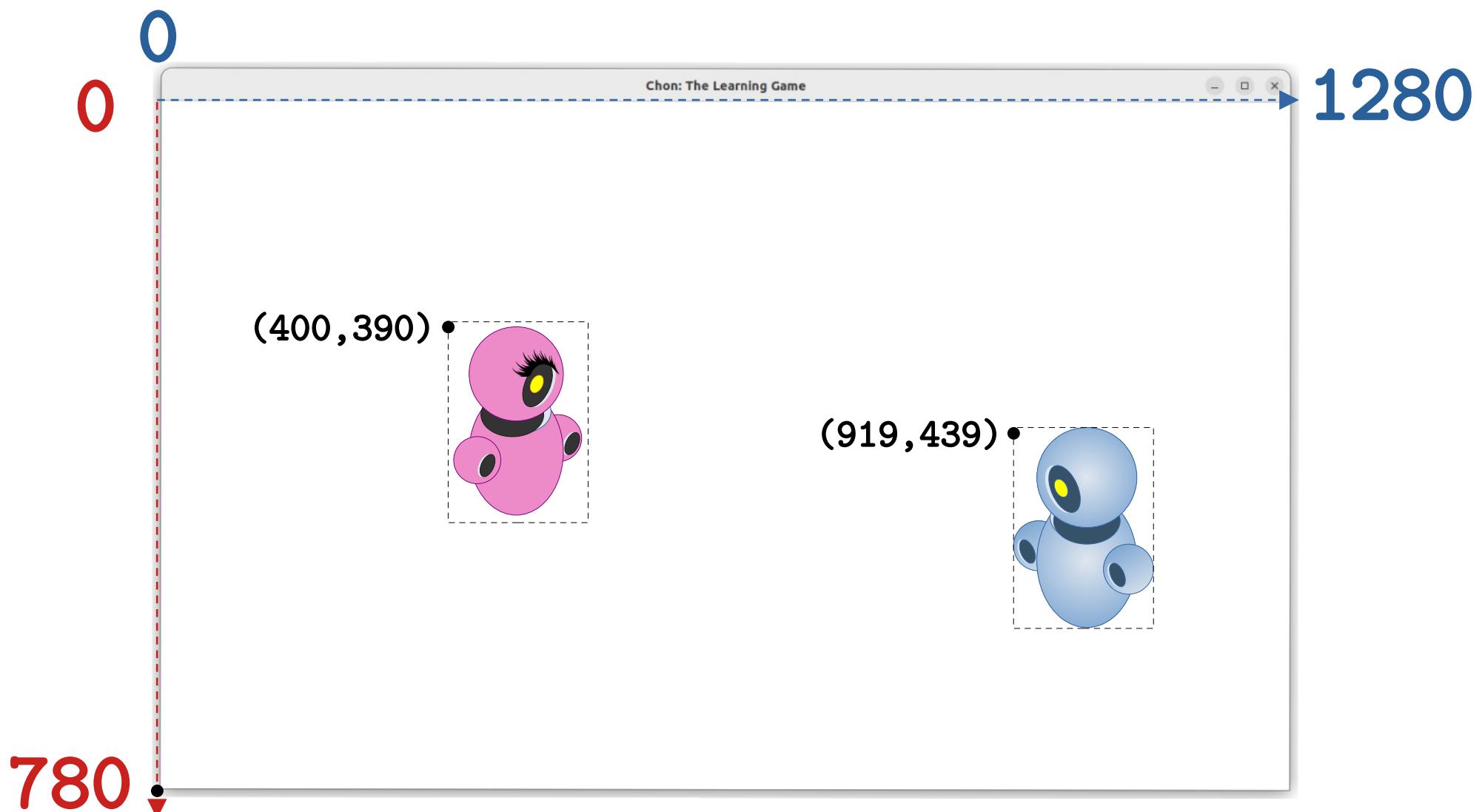
# Chasing the Player

The **two conditions**  
for **x** and **y**  
apply at the same time.

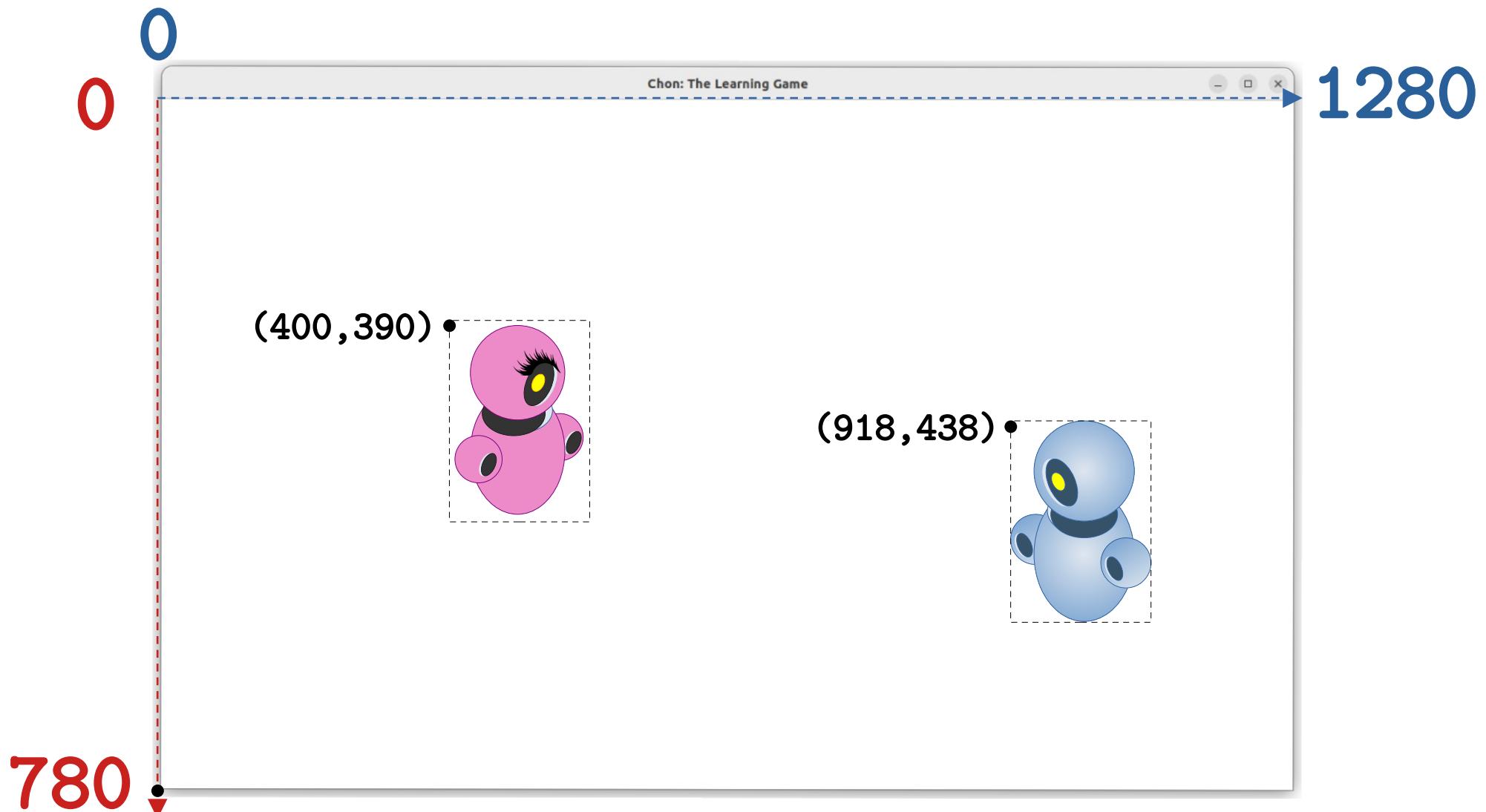
# Chasing the Player



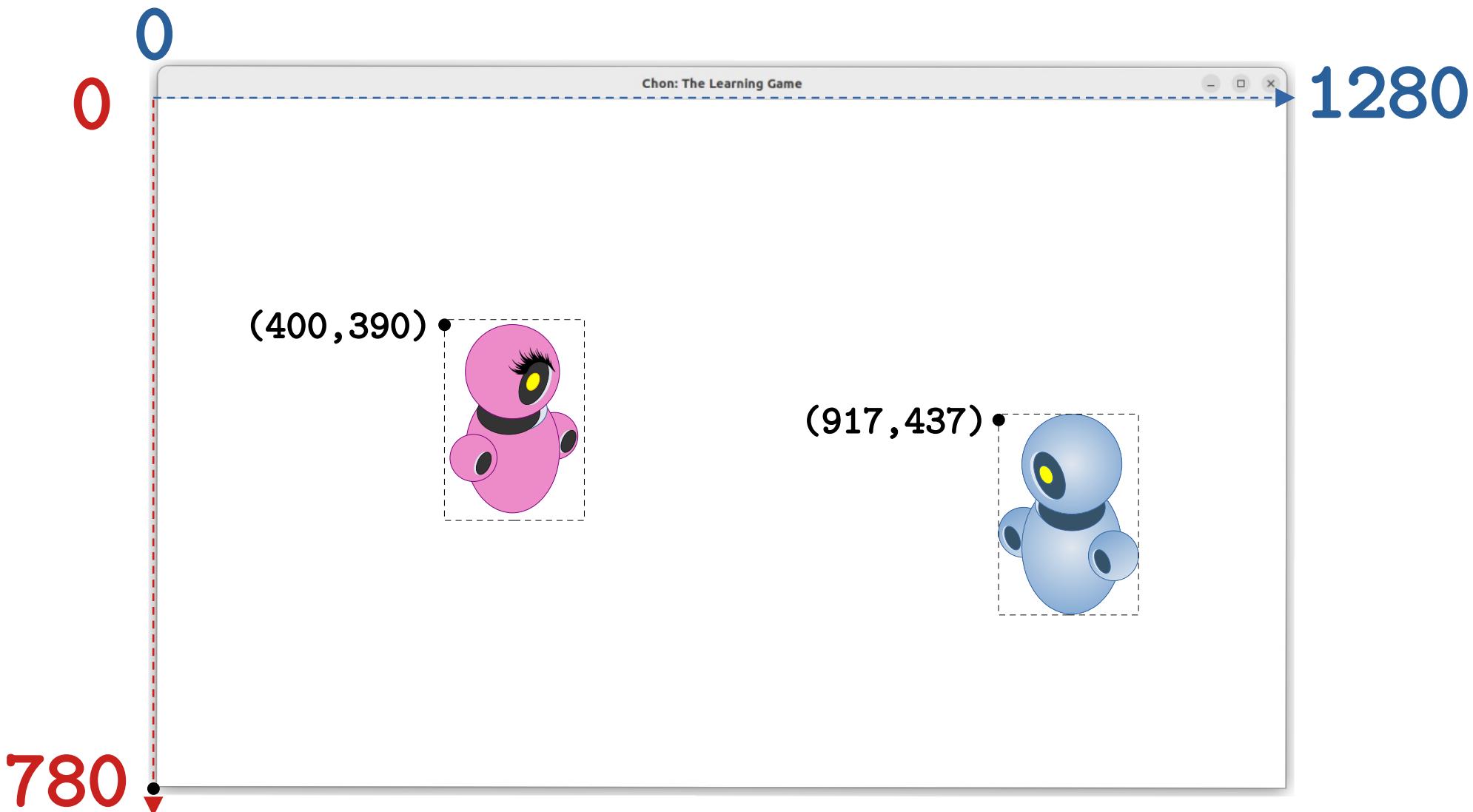
# Chasing the Player



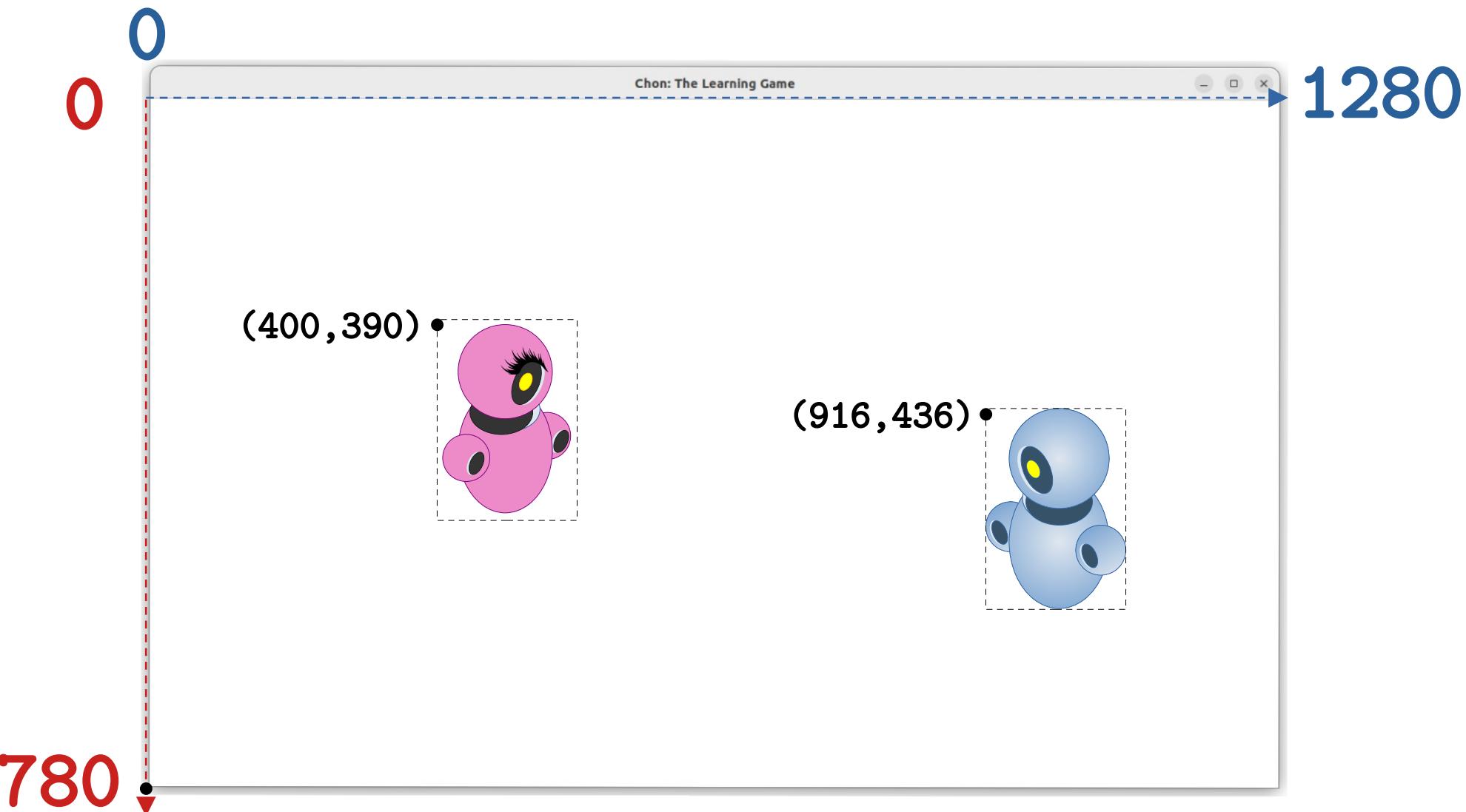
# Chasing the Player



# Chasing the Player



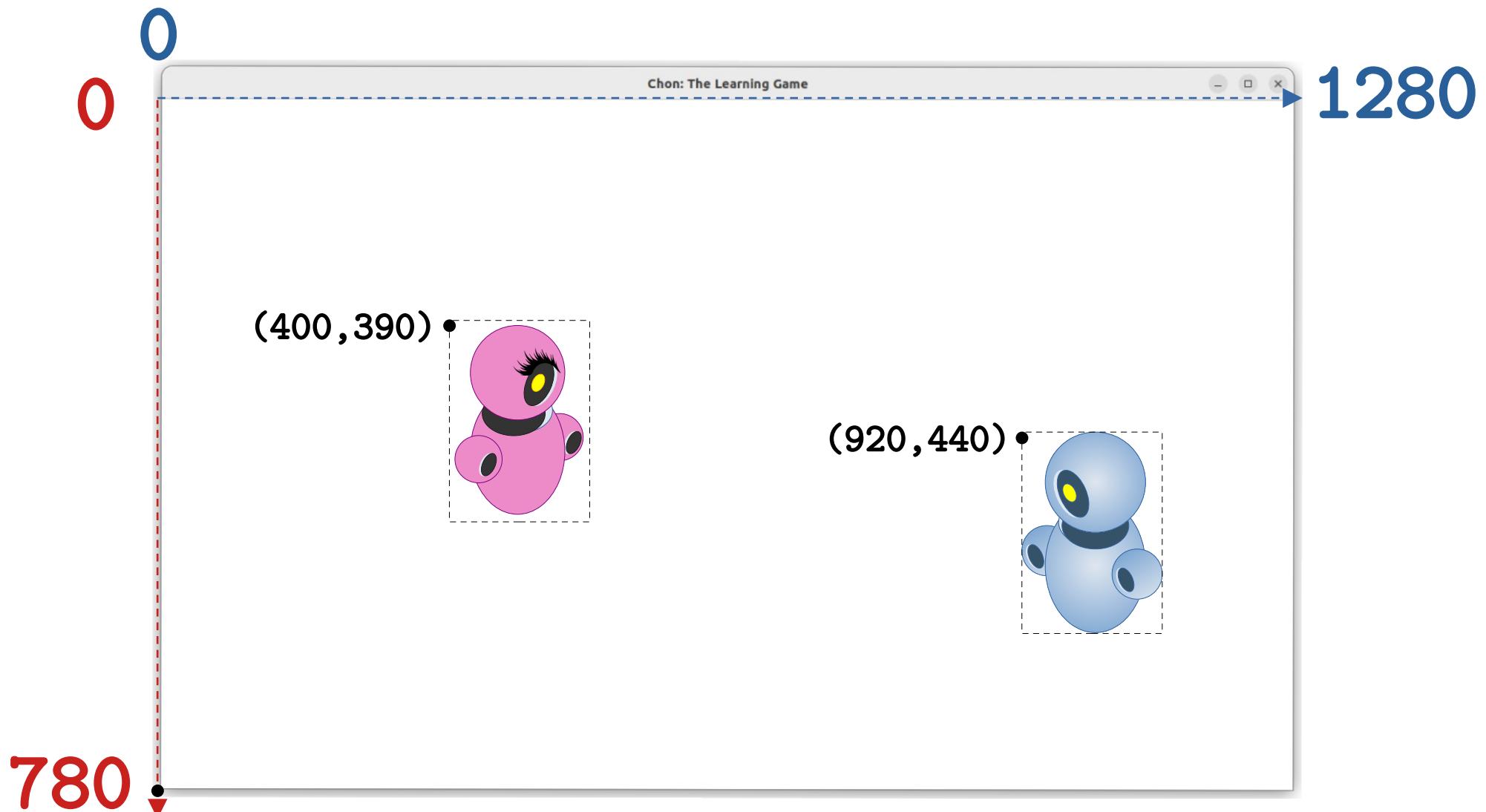
# Chasing the Player



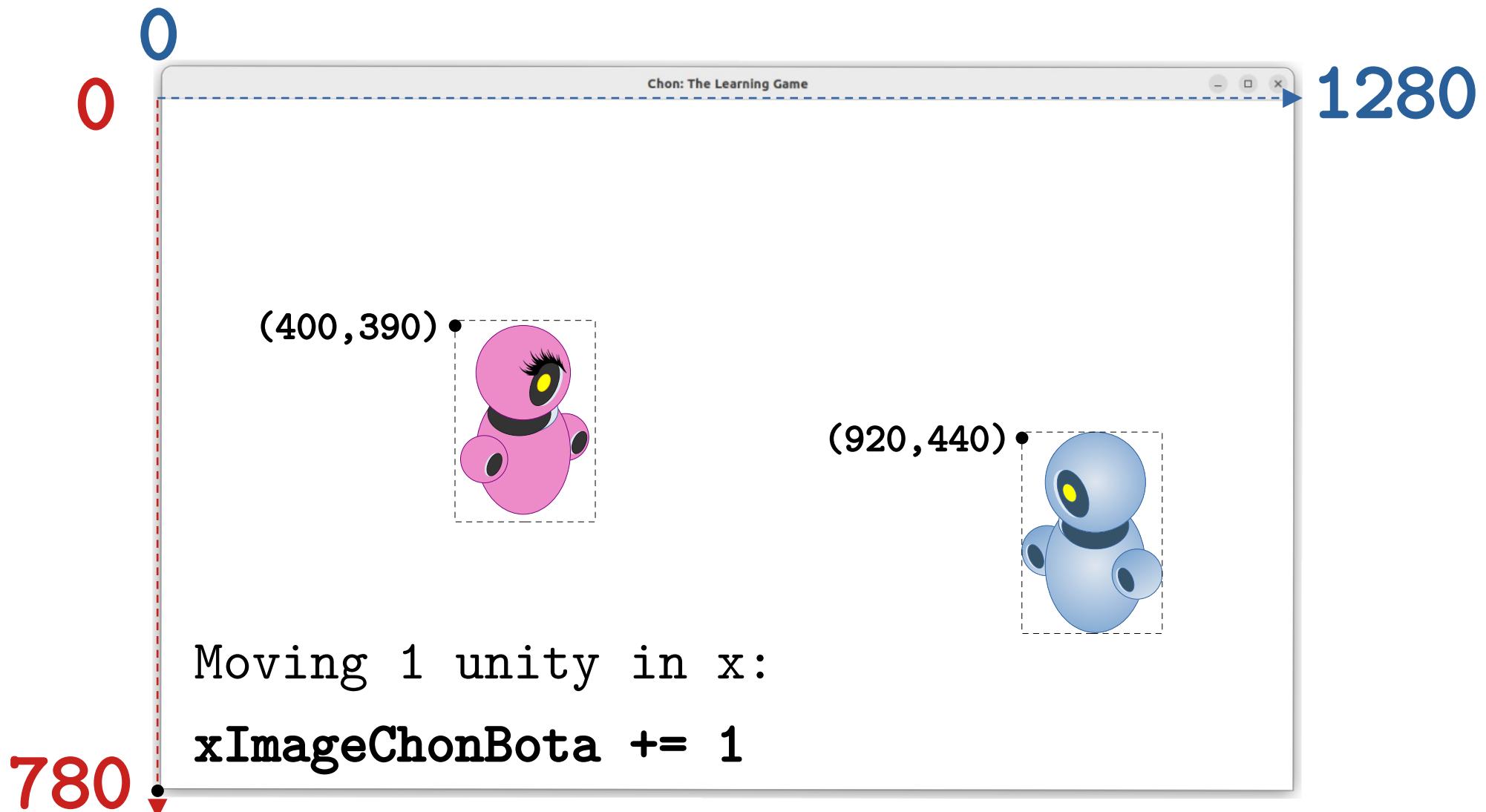
# Defining Speed

The **speed** is related to **how many** unities the object moves in **each cycle** of the **game** loop.

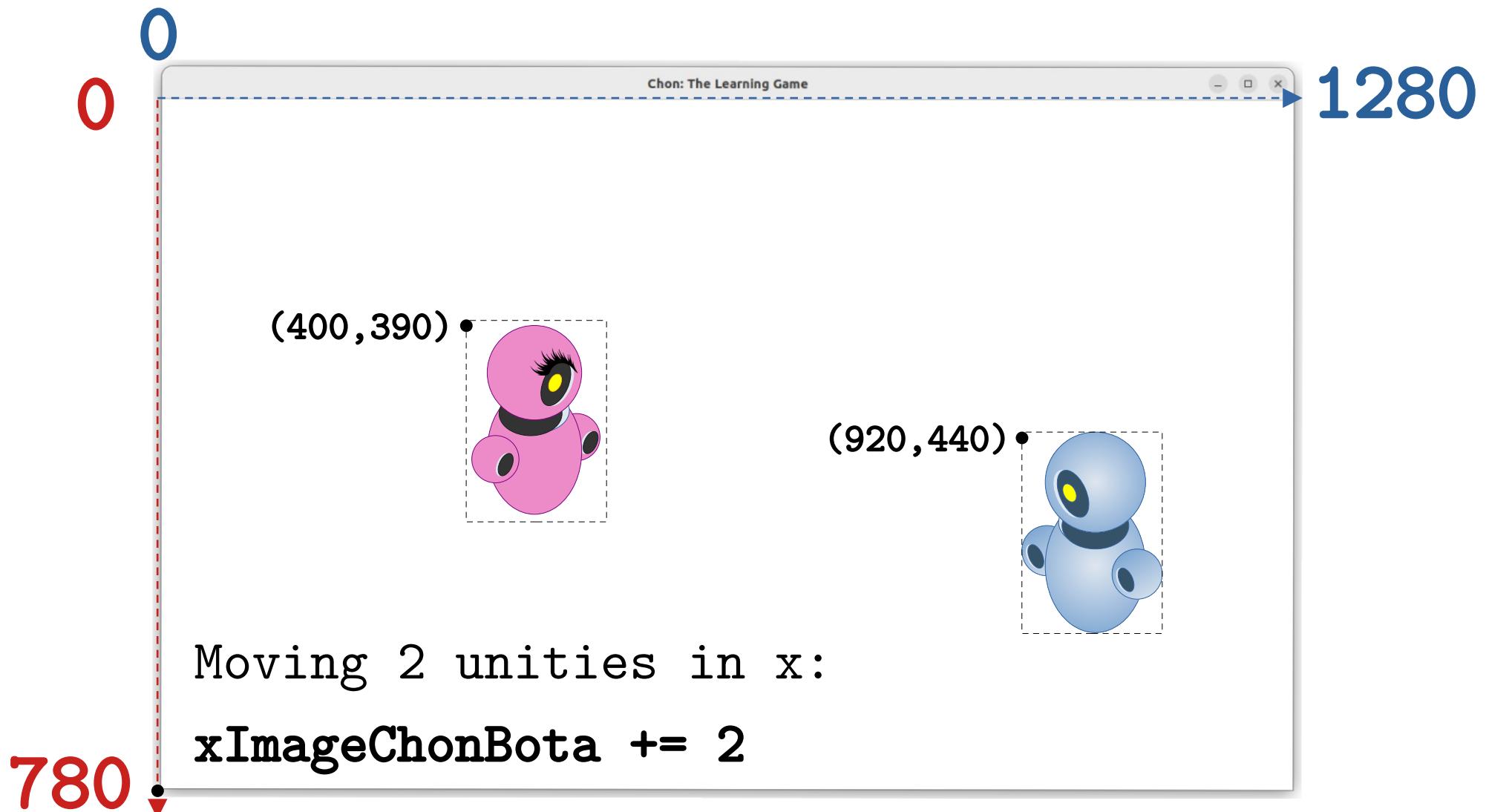
# Defining Speed



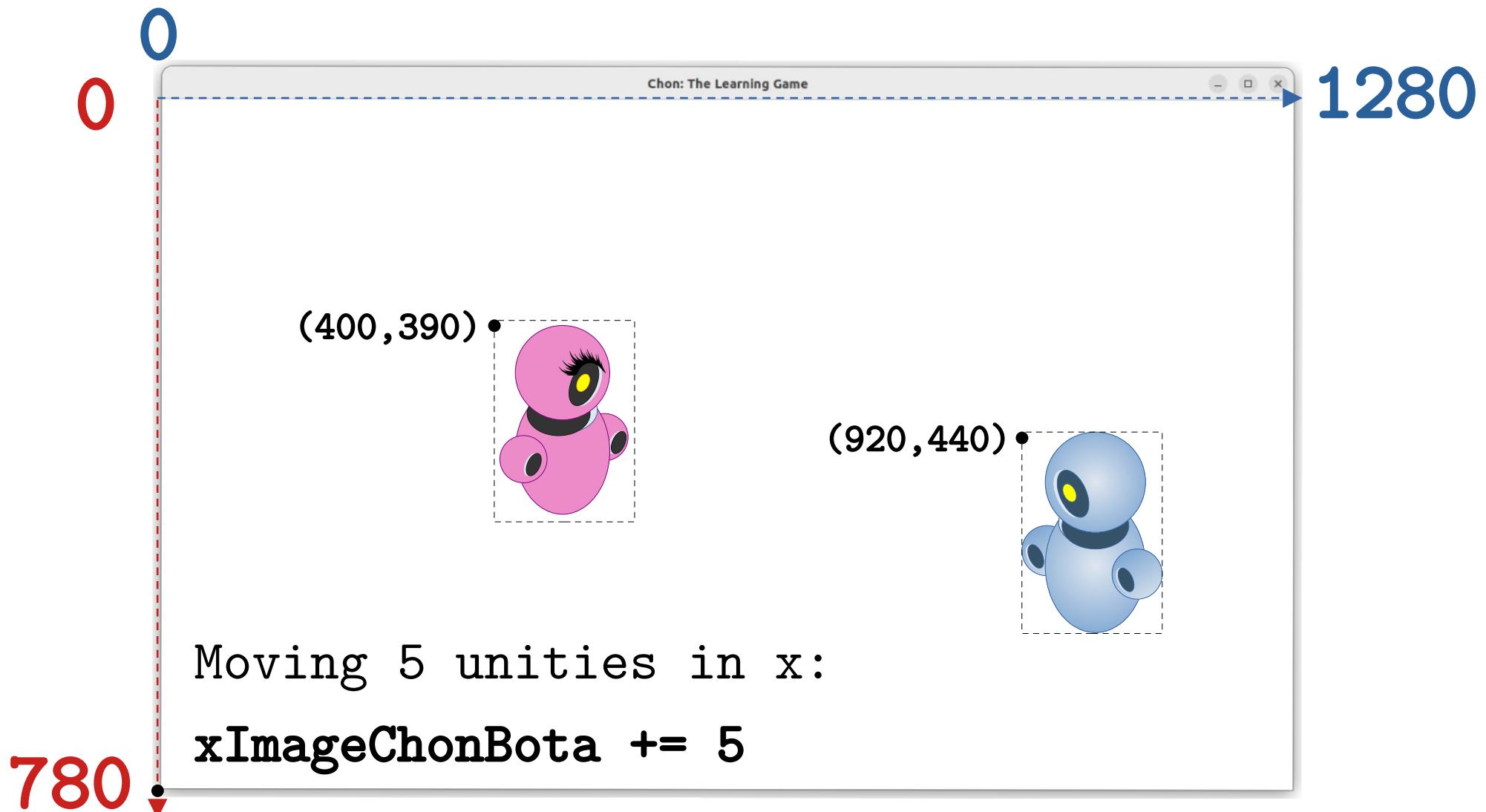
# Defining Speed



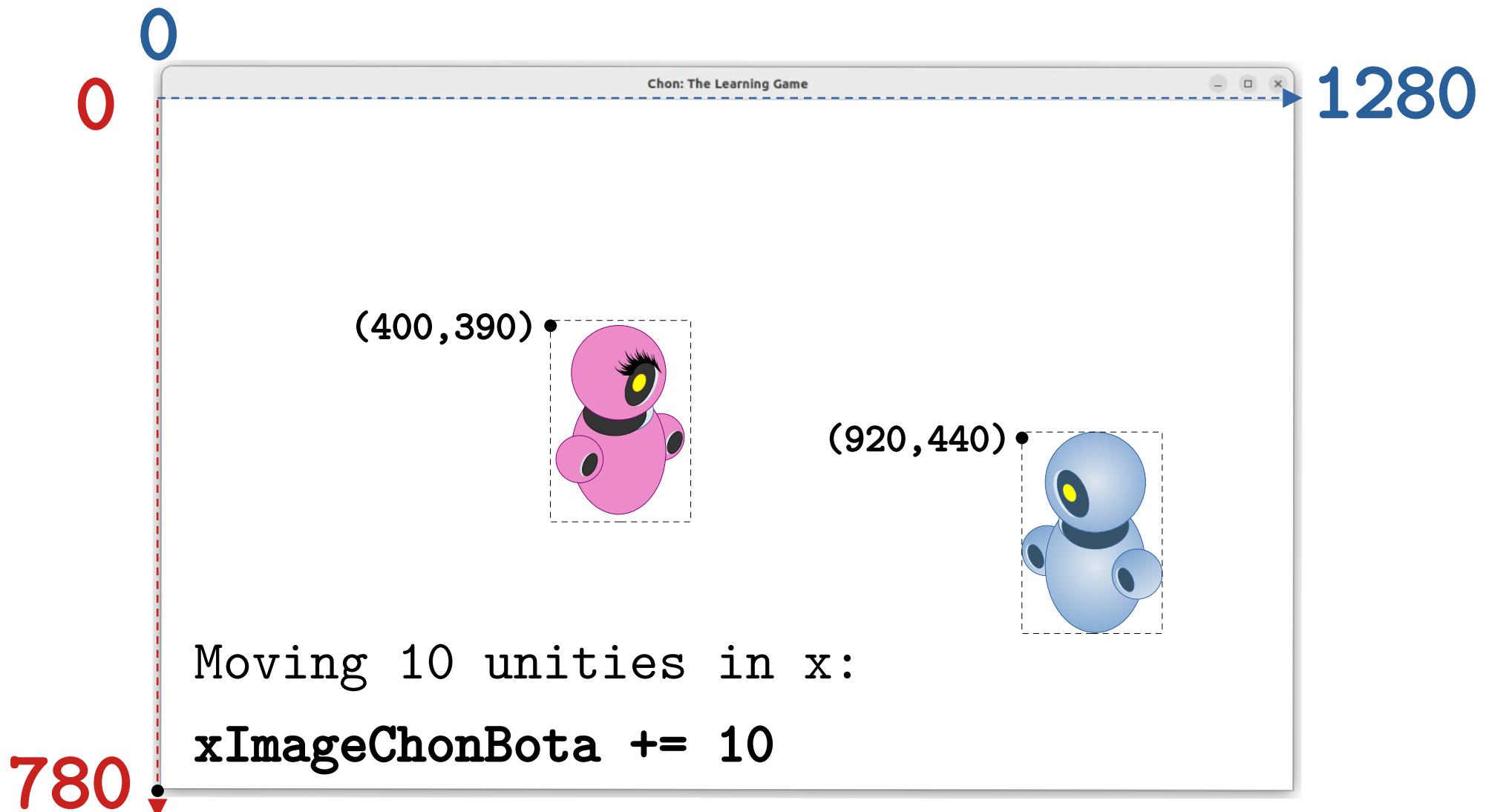
# Defining Speed



# Defining Speed



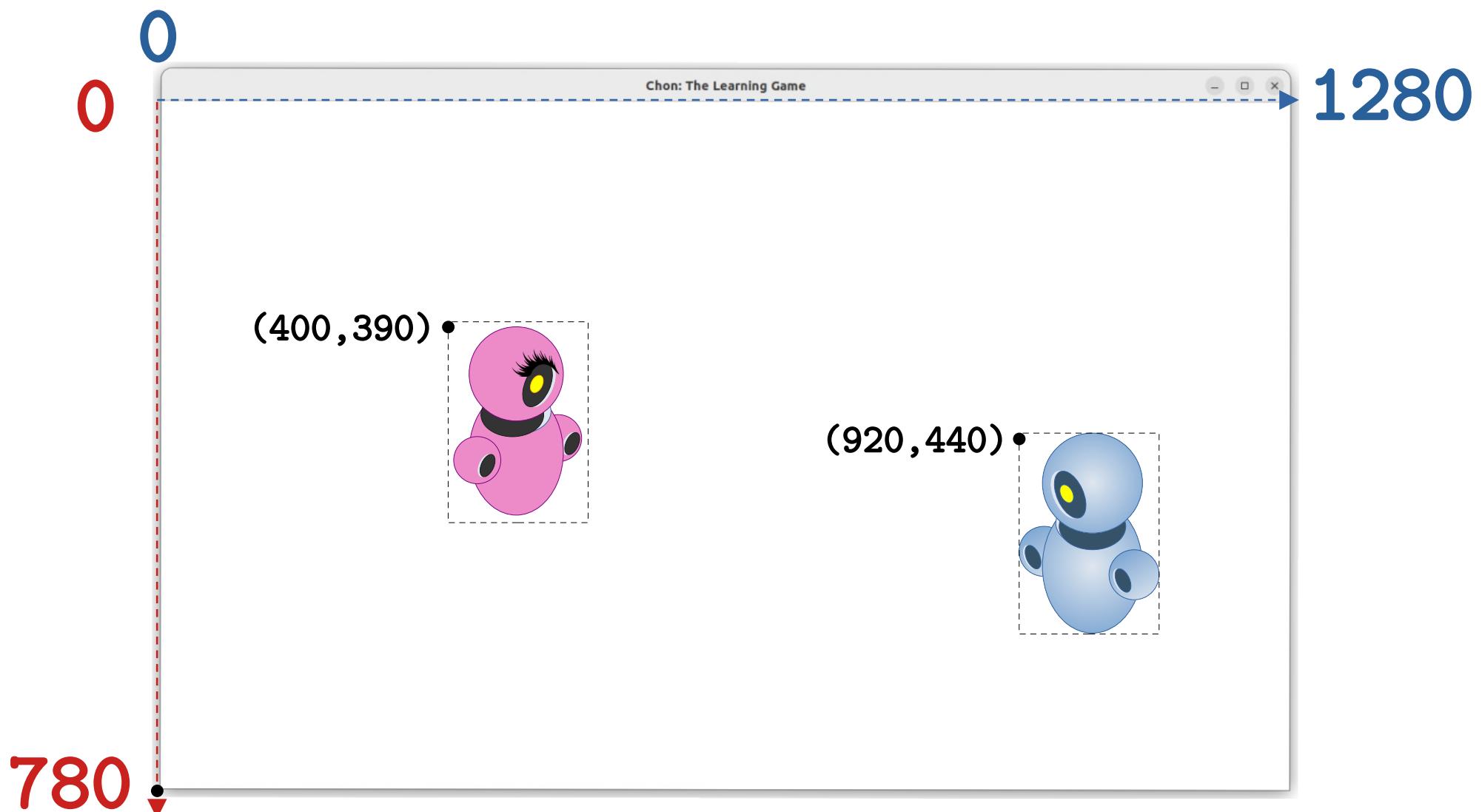
# Defining Speed



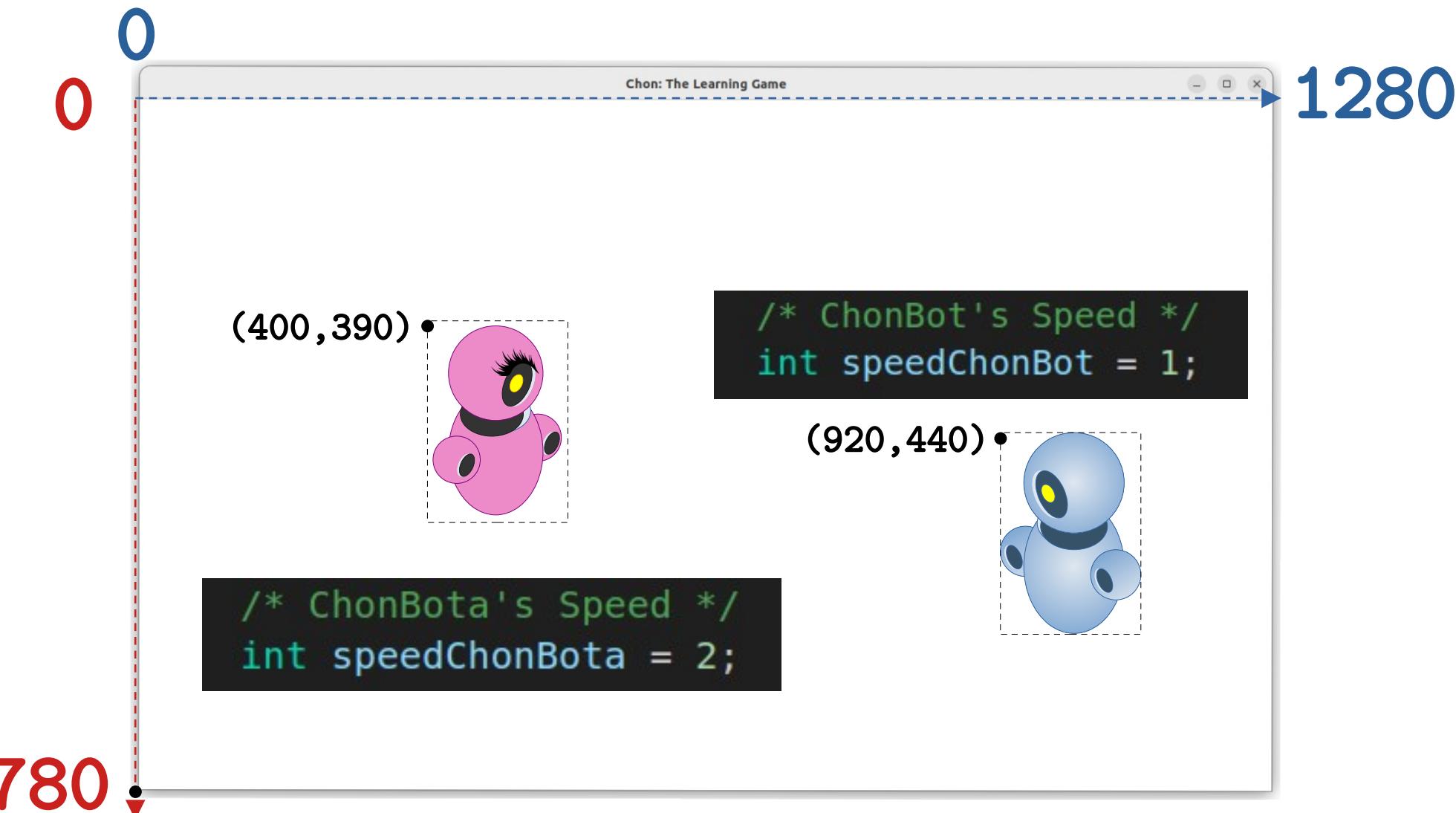
# Defining Speed

So , the **speed** is a pre-defined integer **unity** .

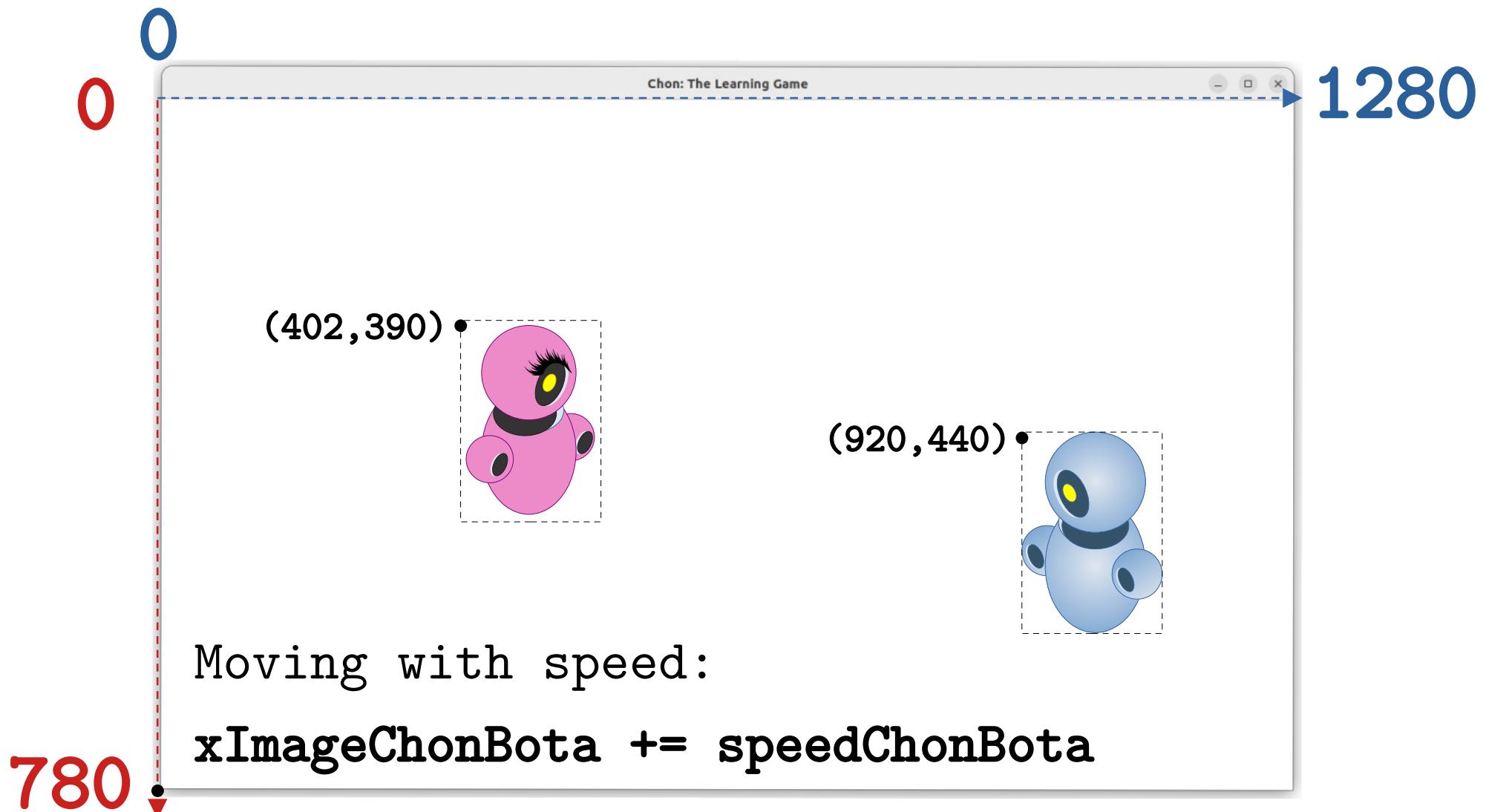
# Defining Speed



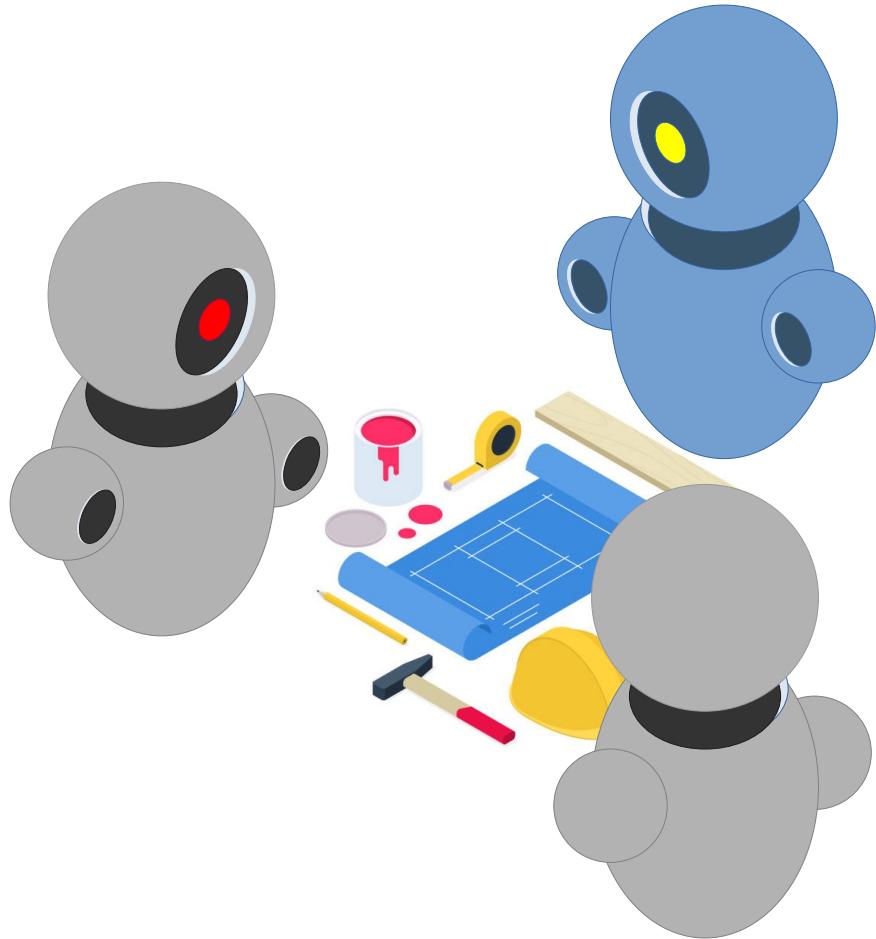
# Defining Speed



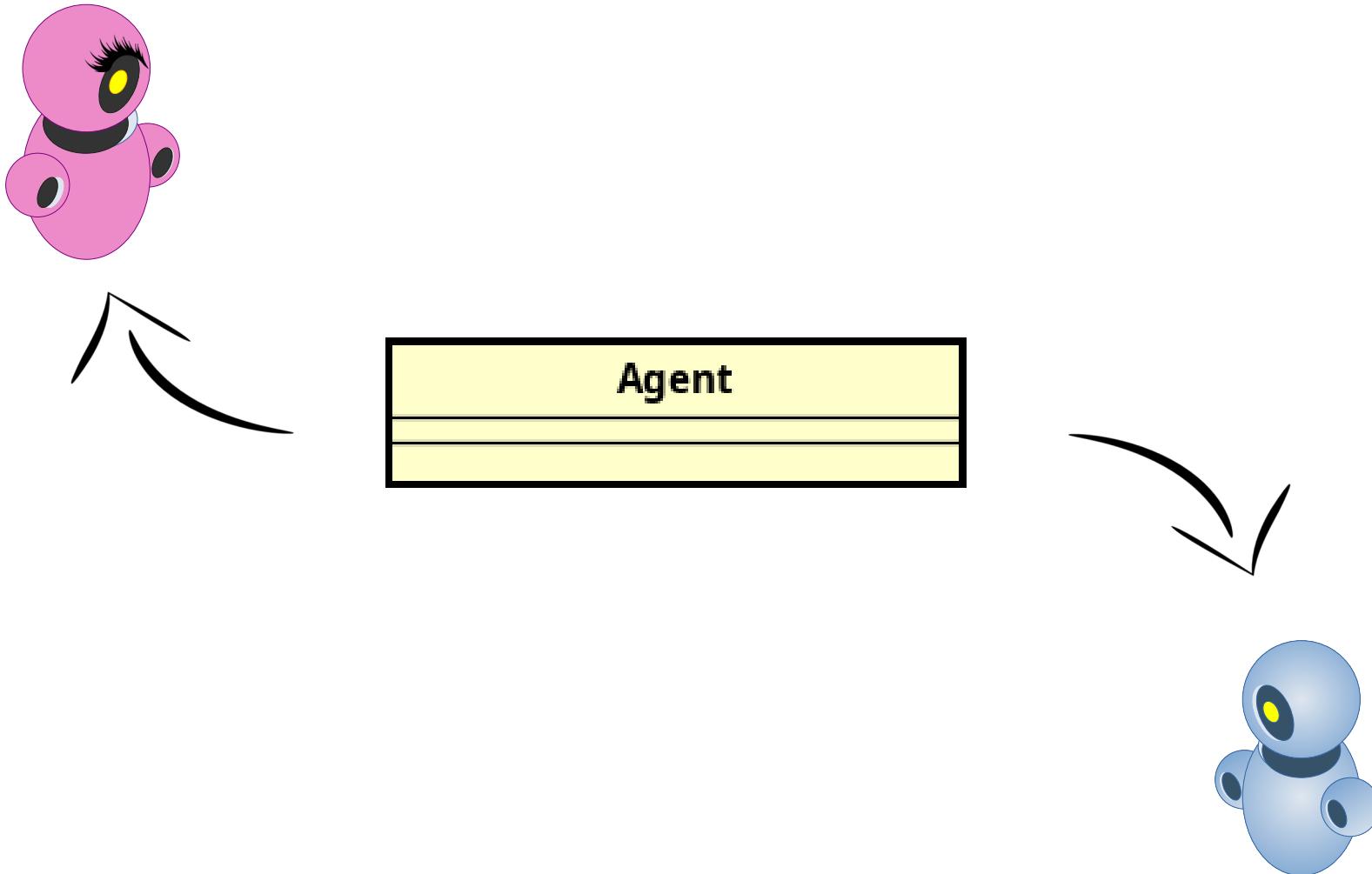
# Defining Speed



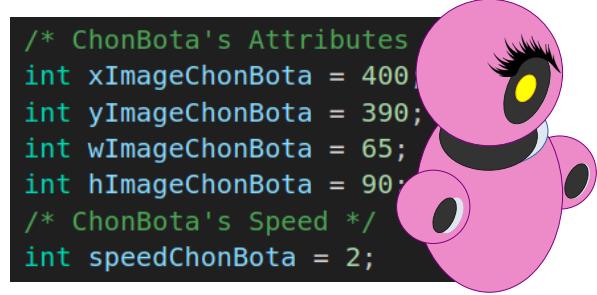
# MODELING THE ENGINE



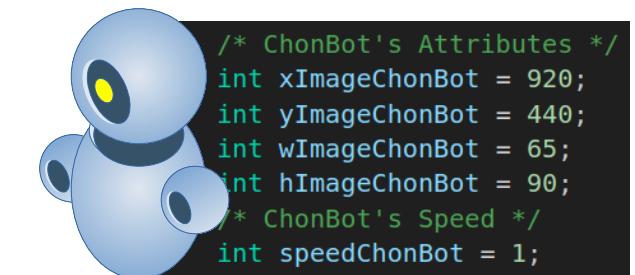
# Class Agent



# Class Agent



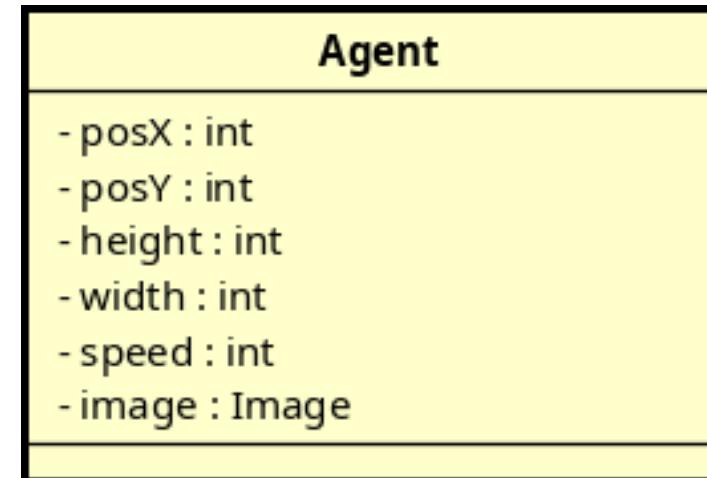
```
/* ChonBota's Attributes  
int xImageChonBota = 400;  
int yImageChonBota = 390;  
int wImageChonBota = 65;  
int hImageChonBota = 90;  
/* ChonBota's Speed */  
int speedChonBota = 2;
```



```
/* ChonBot's Attributes */  
int xImageChonBot = 920;  
int yImageChonBot = 440;  
int wImageChonBot = 65;  
int hImageChonBot = 90;  
/* ChonBot's Speed */  
int speedChonBot = 1;
```

# Class Agent

```
/* ChonBota's Attributes  
int xImageChonBota = 400;  
int yImageChonBota = 390;  
int wImageChonBota = 65;  
int hImageChonBota = 90;  
/* ChonBota's Speed */  
int speedChonBota = 2;
```



```
/* ChonBot's Attributes */  
int xImageChonBot = 920;  
int yImageChonBot = 440;  
int wImageChonBot = 65;  
int hImageChonBot = 90;  
/* ChonBot's Speed */  
int speedChonBot = 1;
```

# Class Environment

```
StackPane root = new StackPane();
Scene scene = new Scene(root, width:1280, height:780);
theStage.setTitle(value:"Chon: The Learning Game");
theStage.setScene(scene);

int wCanvas = 1280;
int hCanvas = 780;
Canvas canvas = new Canvas(wCanvas, hCanvas);
```

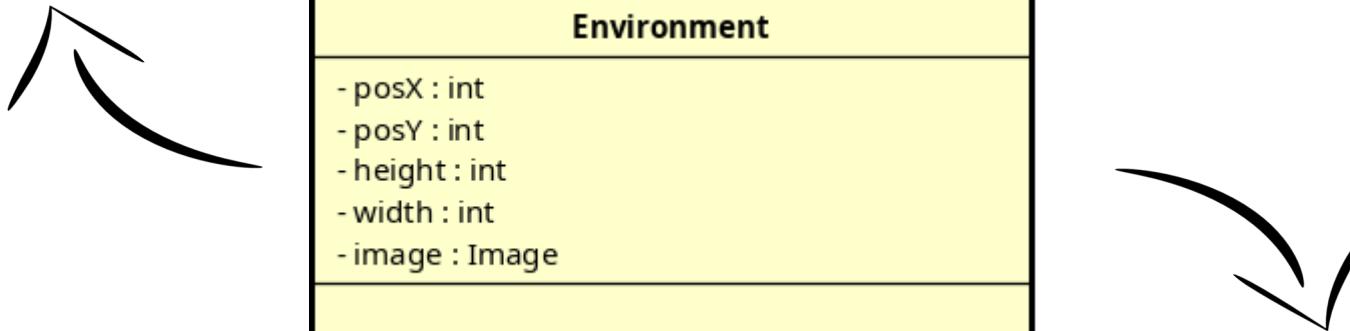


```
Image background = new Image(getClass().getResource("/images/environment/castle.png").toExternalForm());
```

# Class Environment

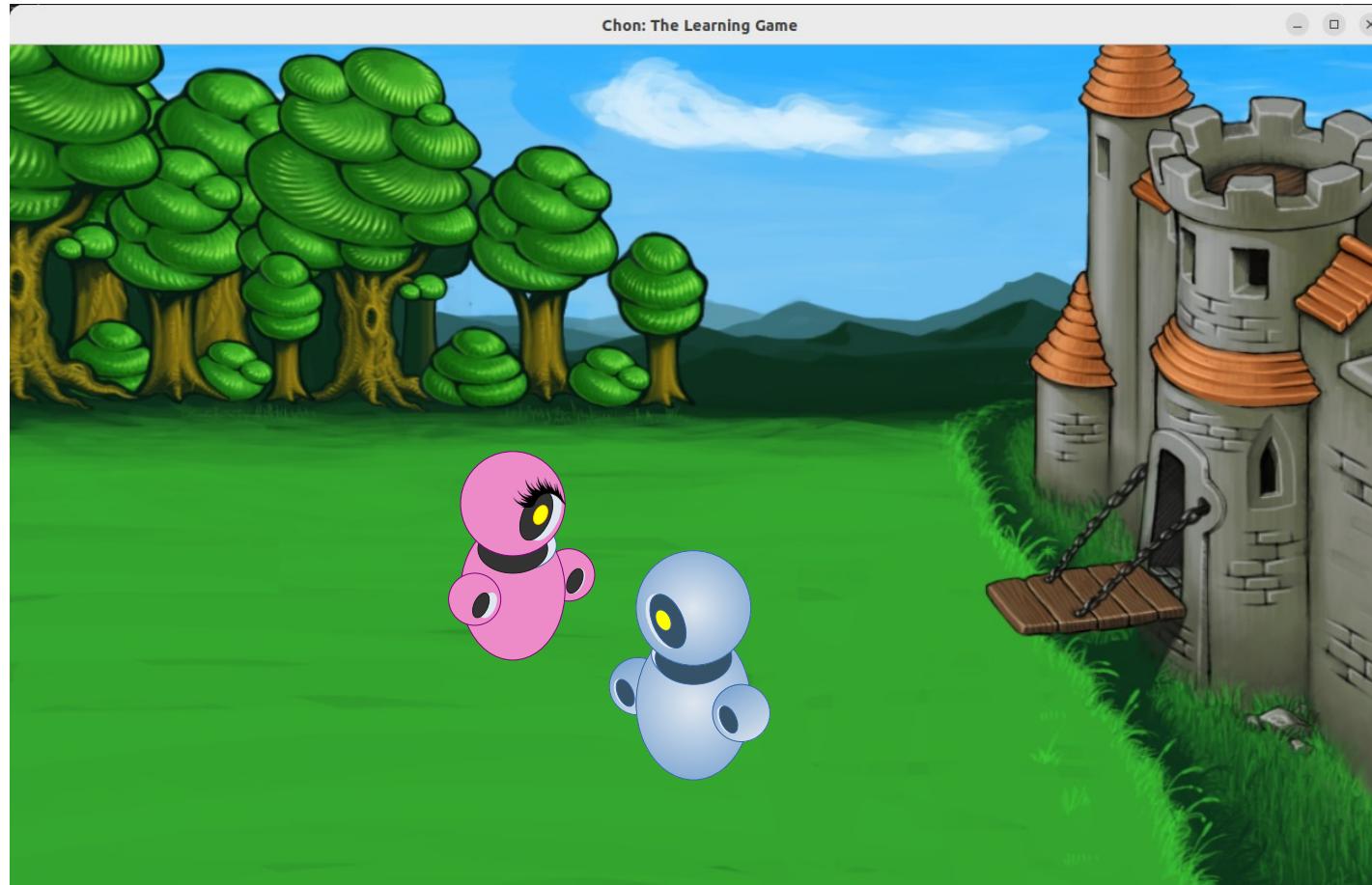
```
StackPane root = new StackPane();
Scene scene = new Scene(root, width:1280, height:780);
theStage.setTitle(value:"Chon: The Learning Game");
theStage.setScene(scene);

int wCanvas = 1280;
int hCanvas = 780;
Canvas canvas = new Canvas(wCanvas, hCanvas);
```

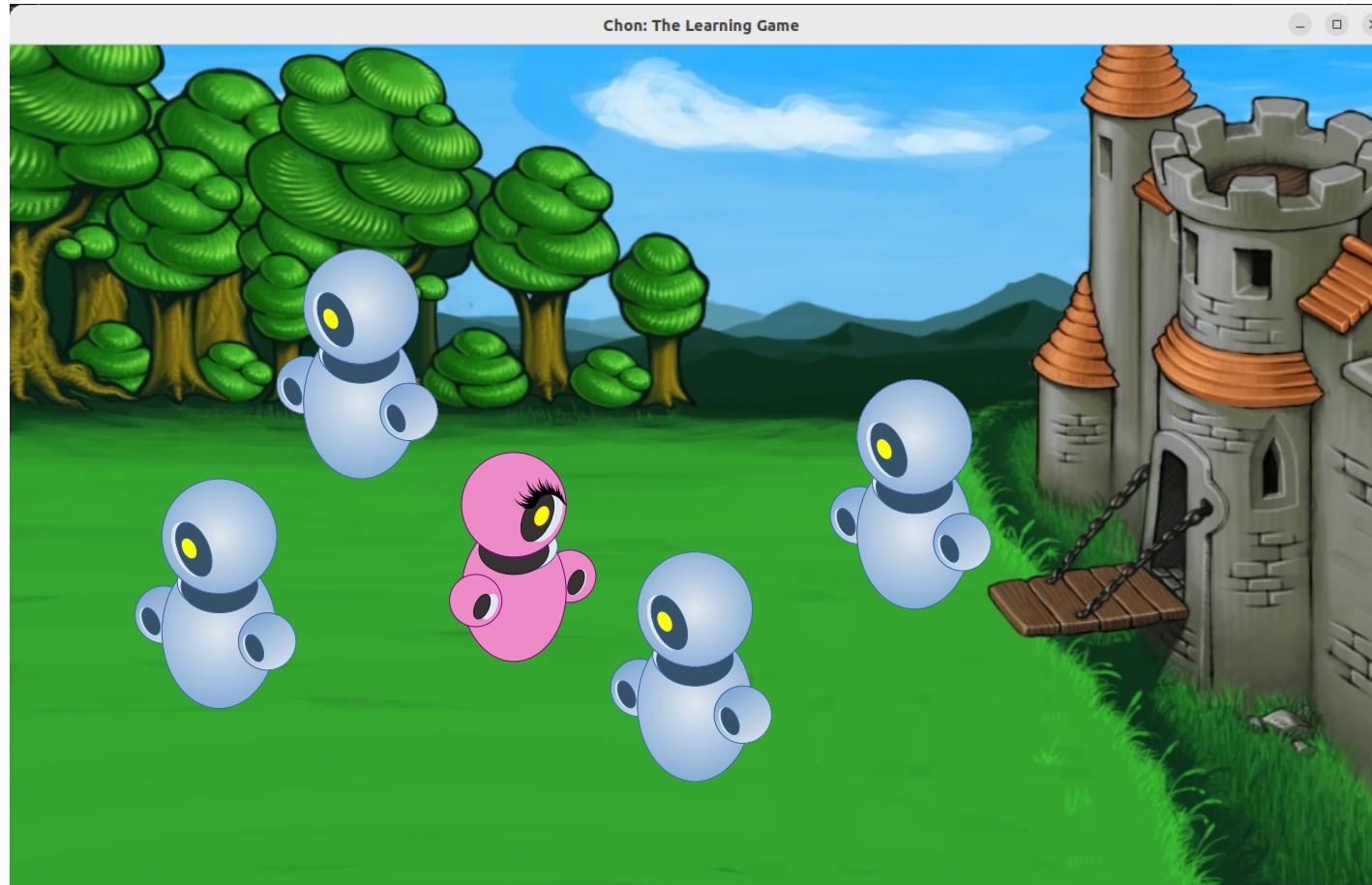


```
Image background = new Image(getClass().getResource("/images/environment/castle.png").toExternalForm());
```

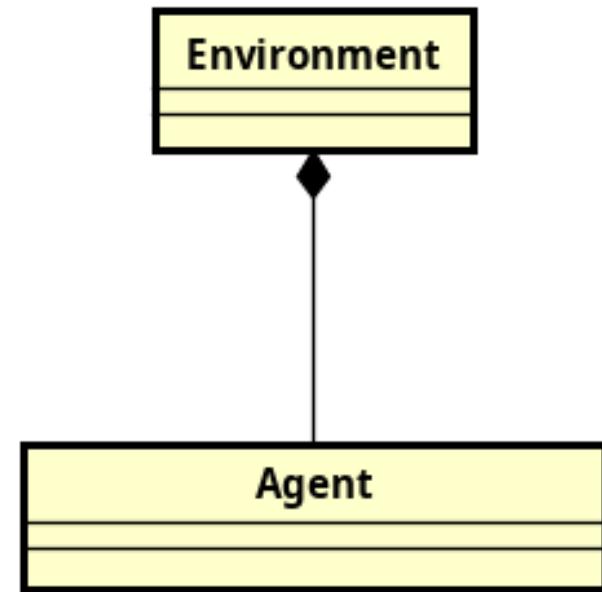
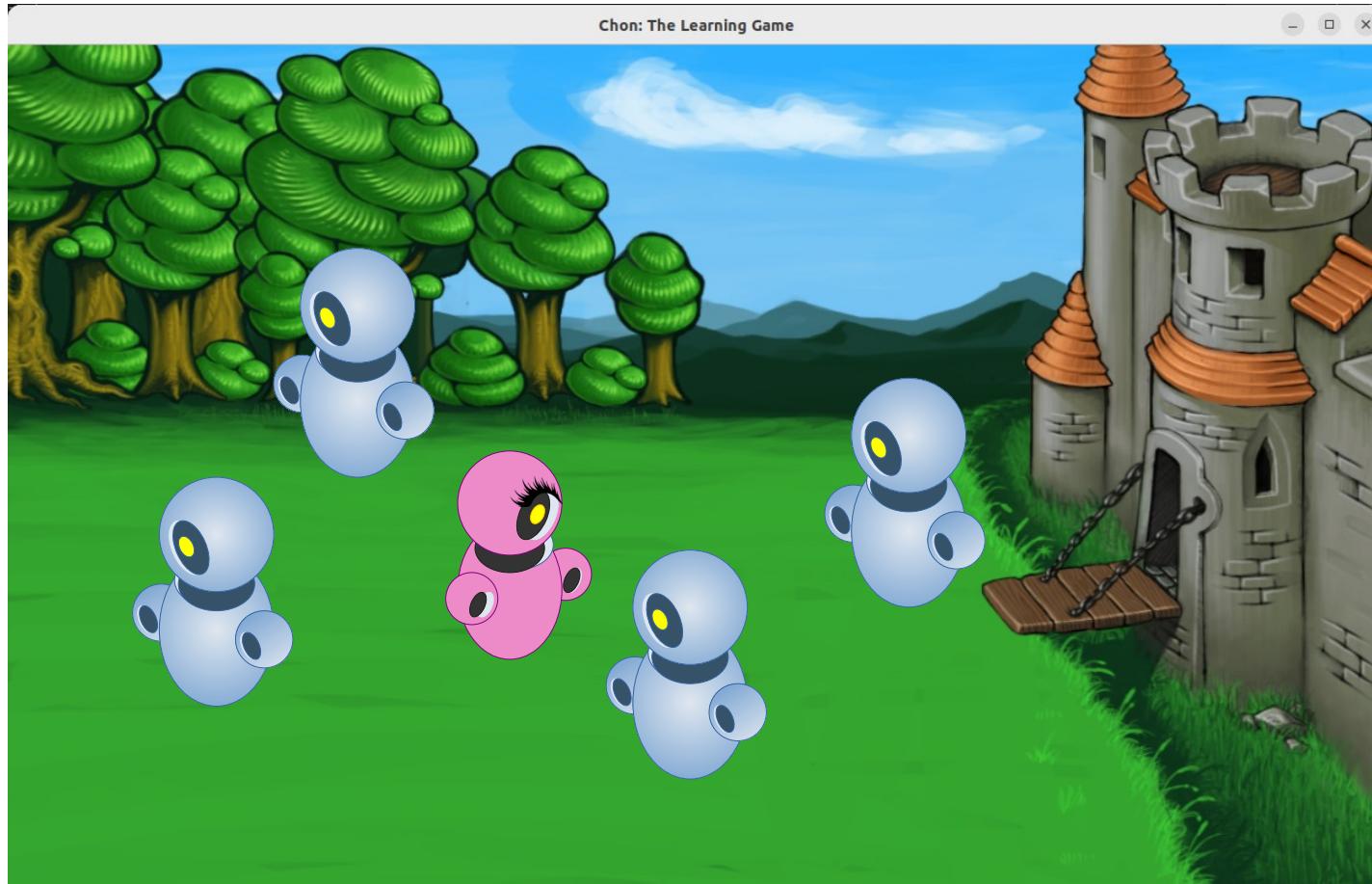
# The Class Diagram



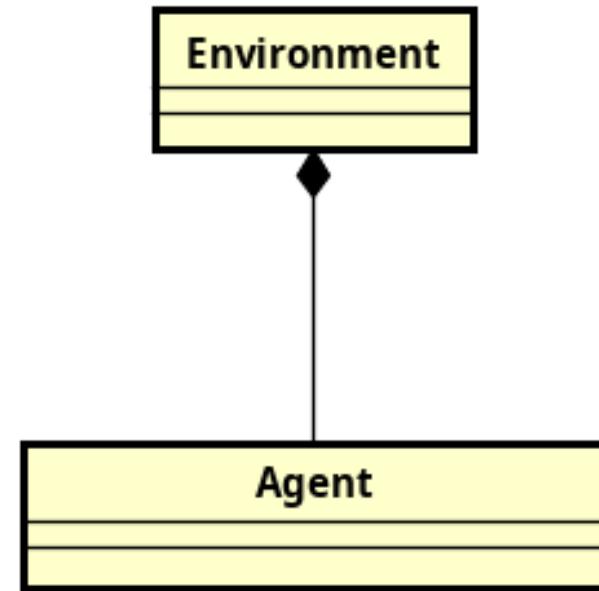
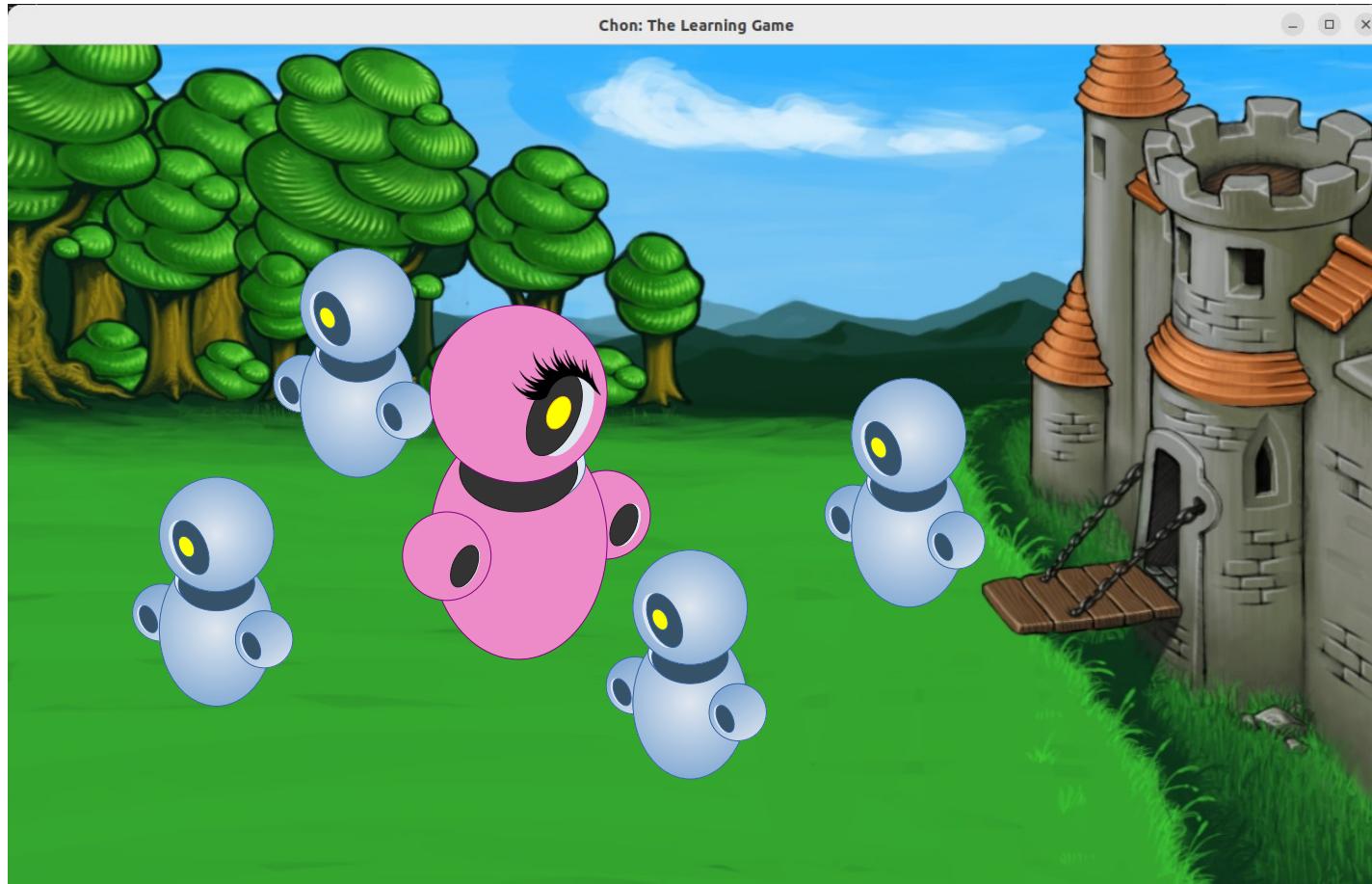
# The Class Diagram



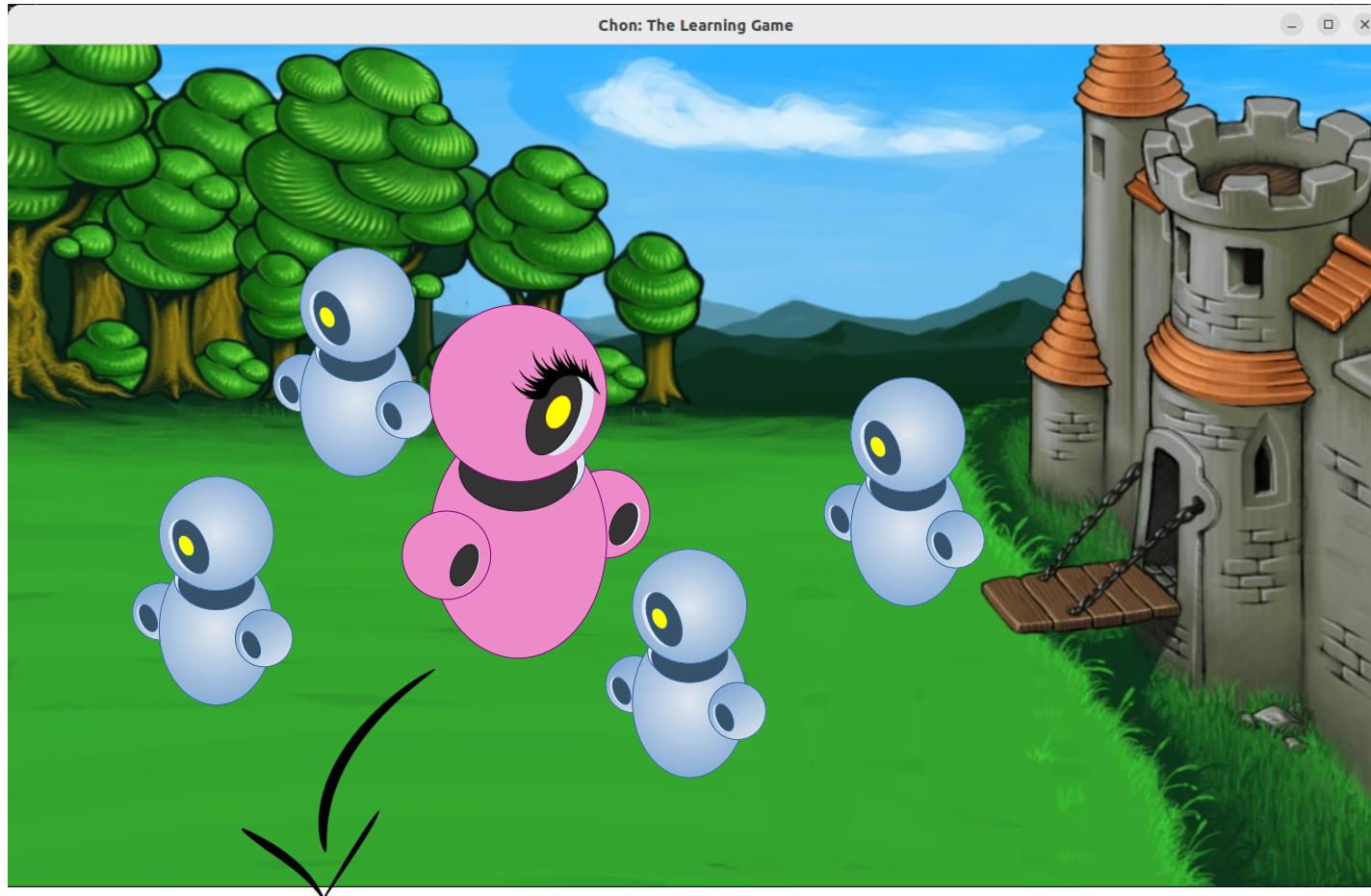
# The Class Diagram



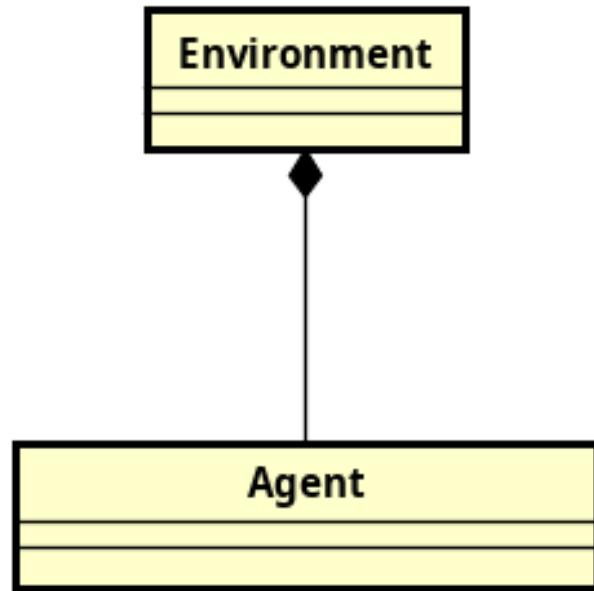
# The Class Diagram



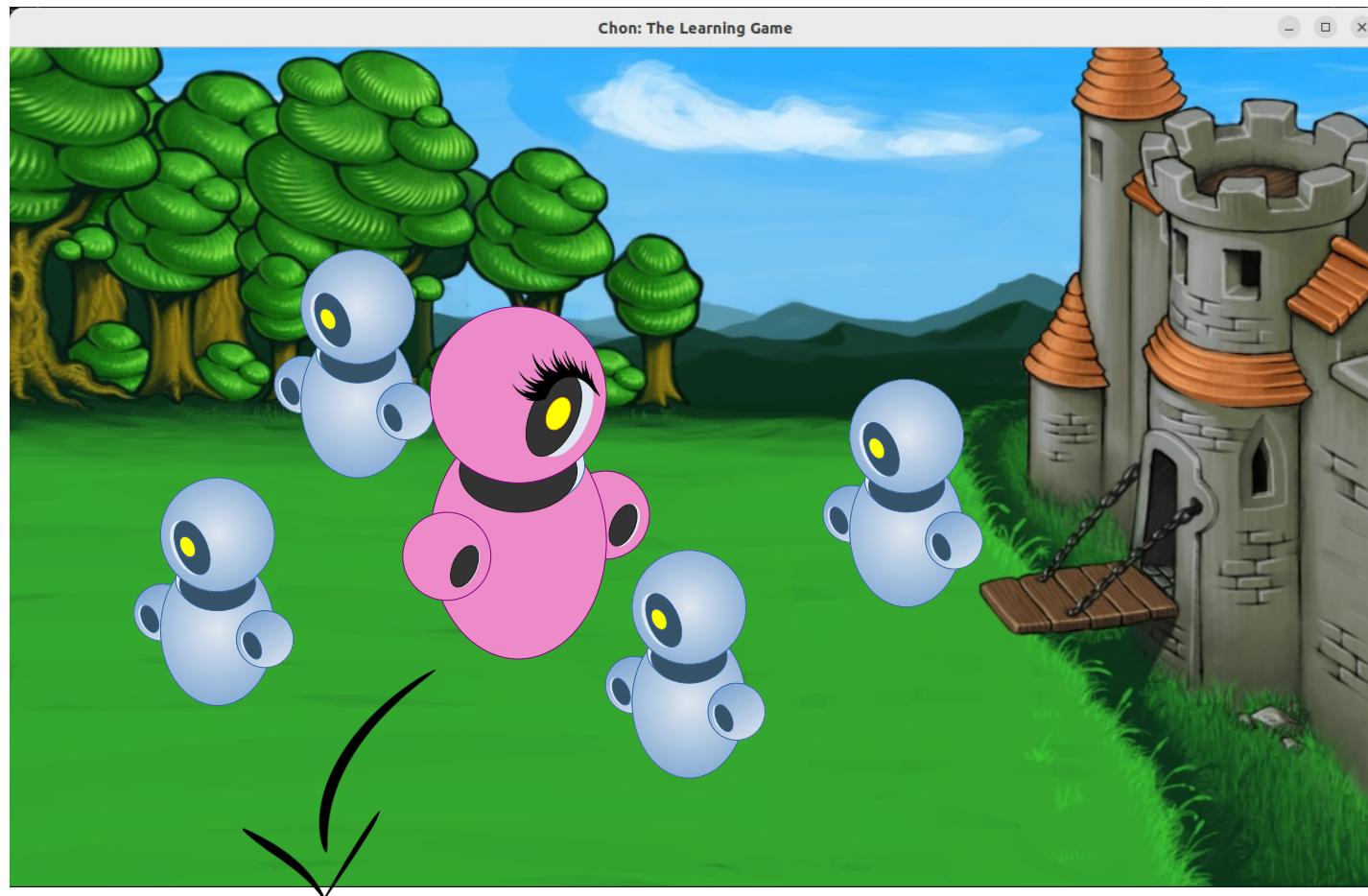
# The Class Diagram



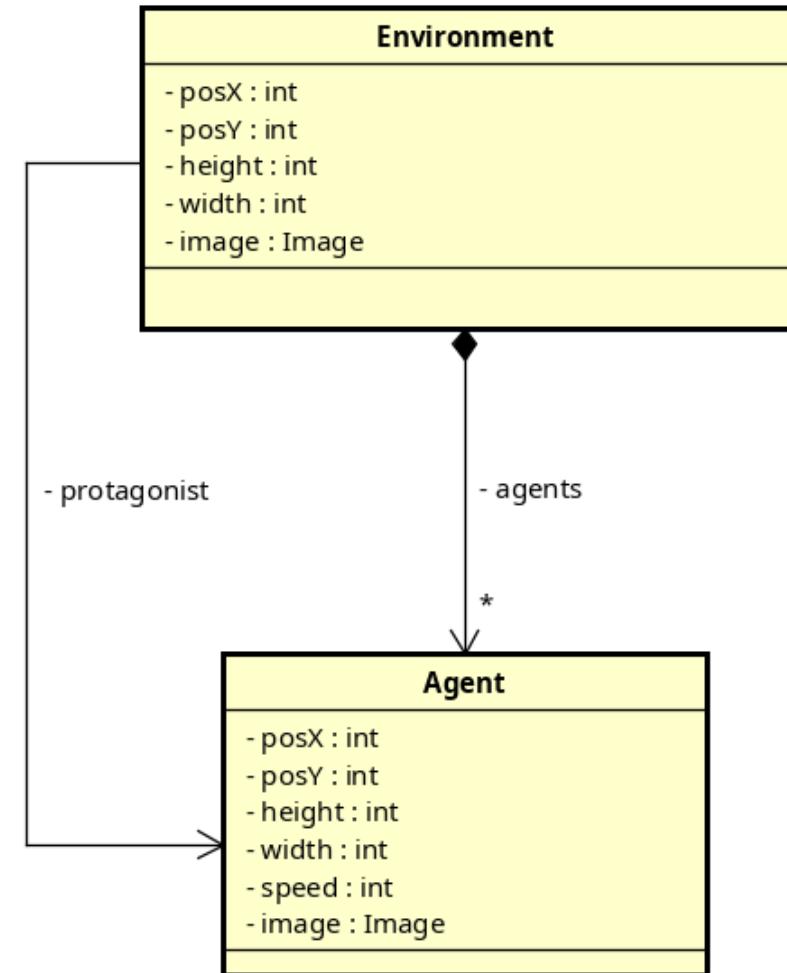
**protagonist**



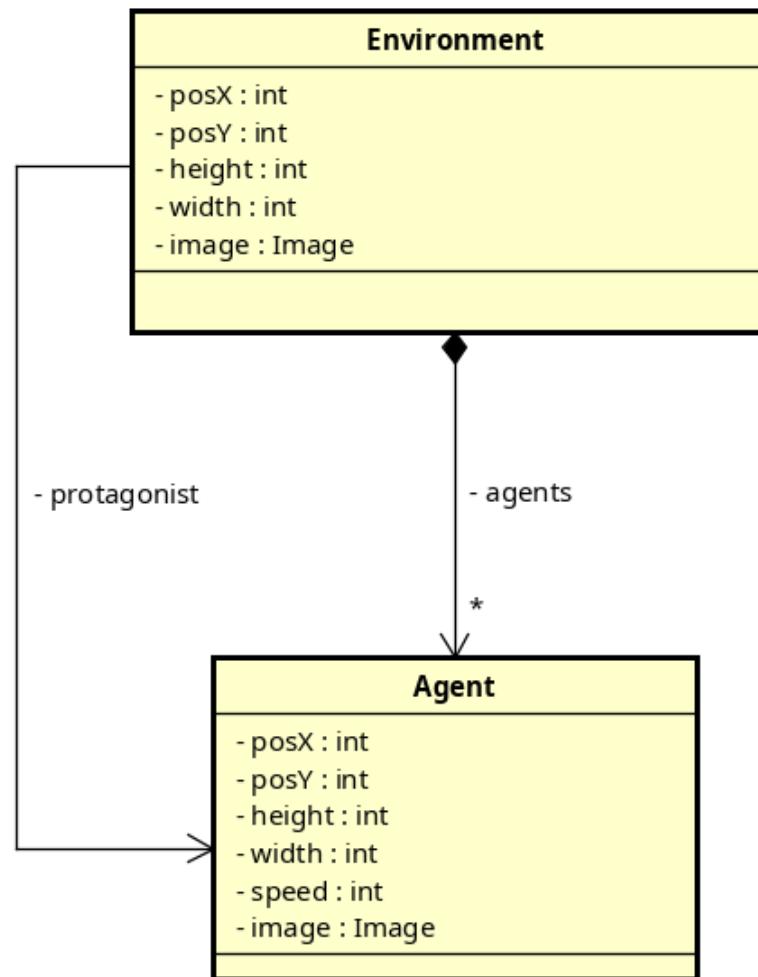
# The Class Diagram



protagonist

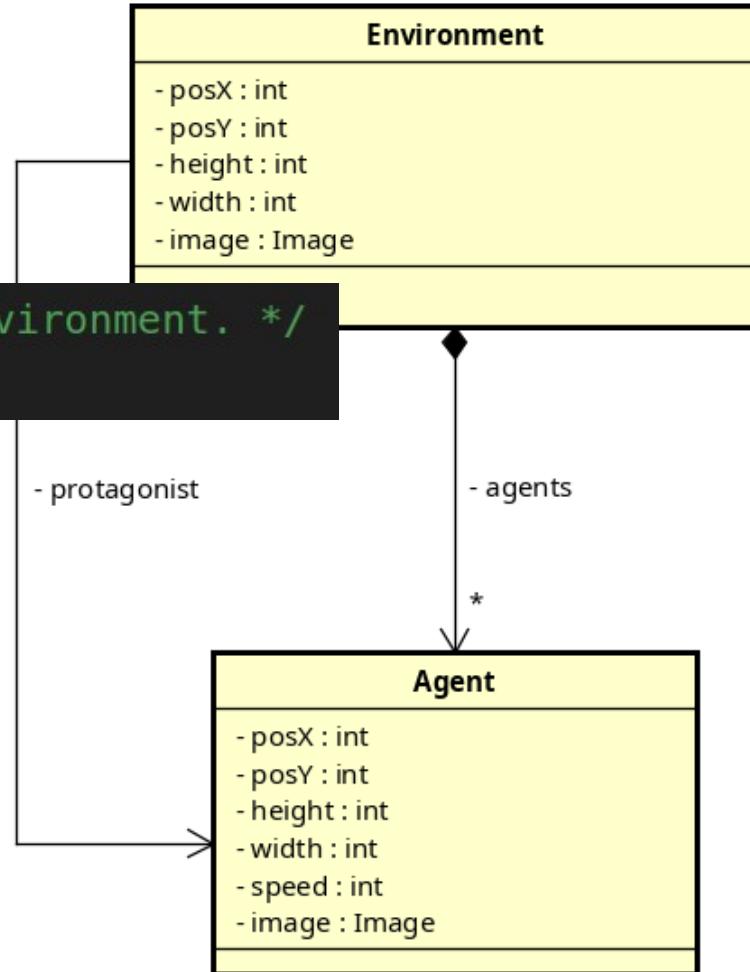


# The Class Diagram

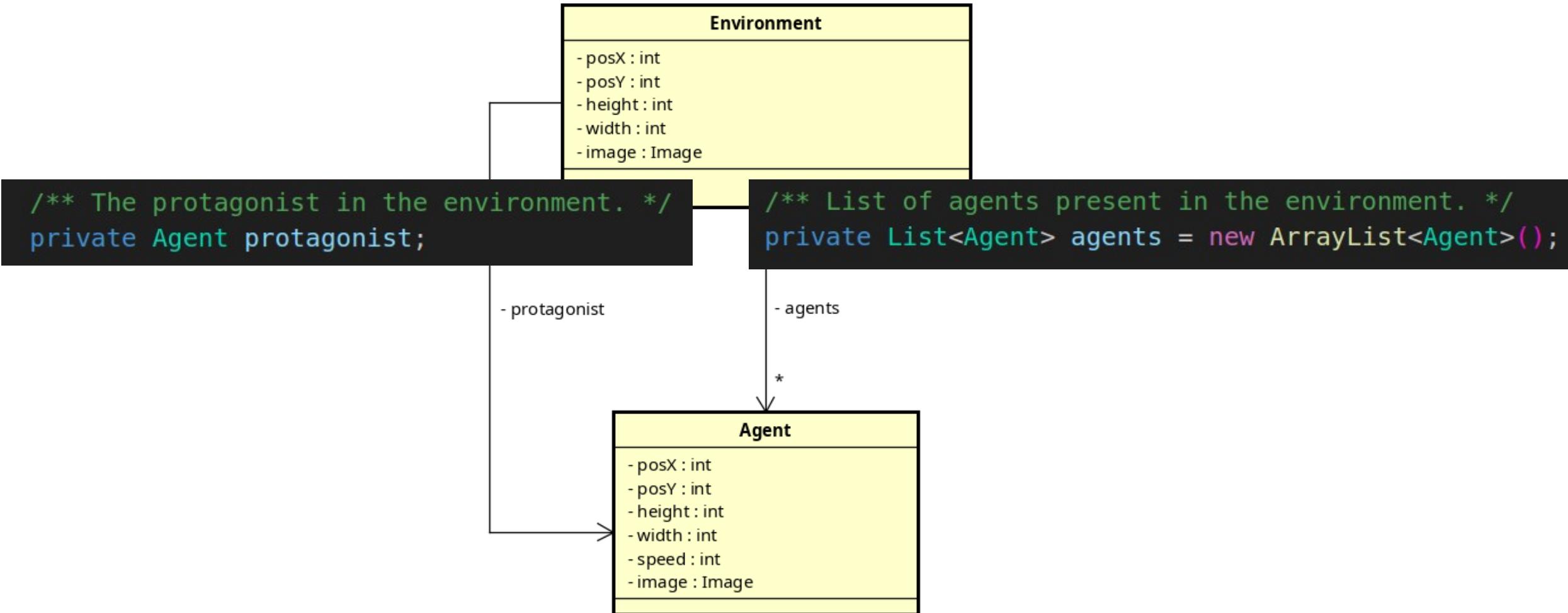


# The Class Diagram

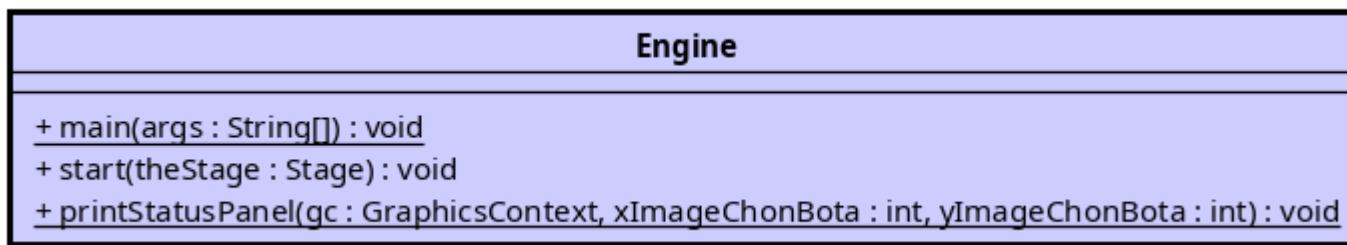
```
/** The protagonist in the environment. */
private Agent protagonist;
```



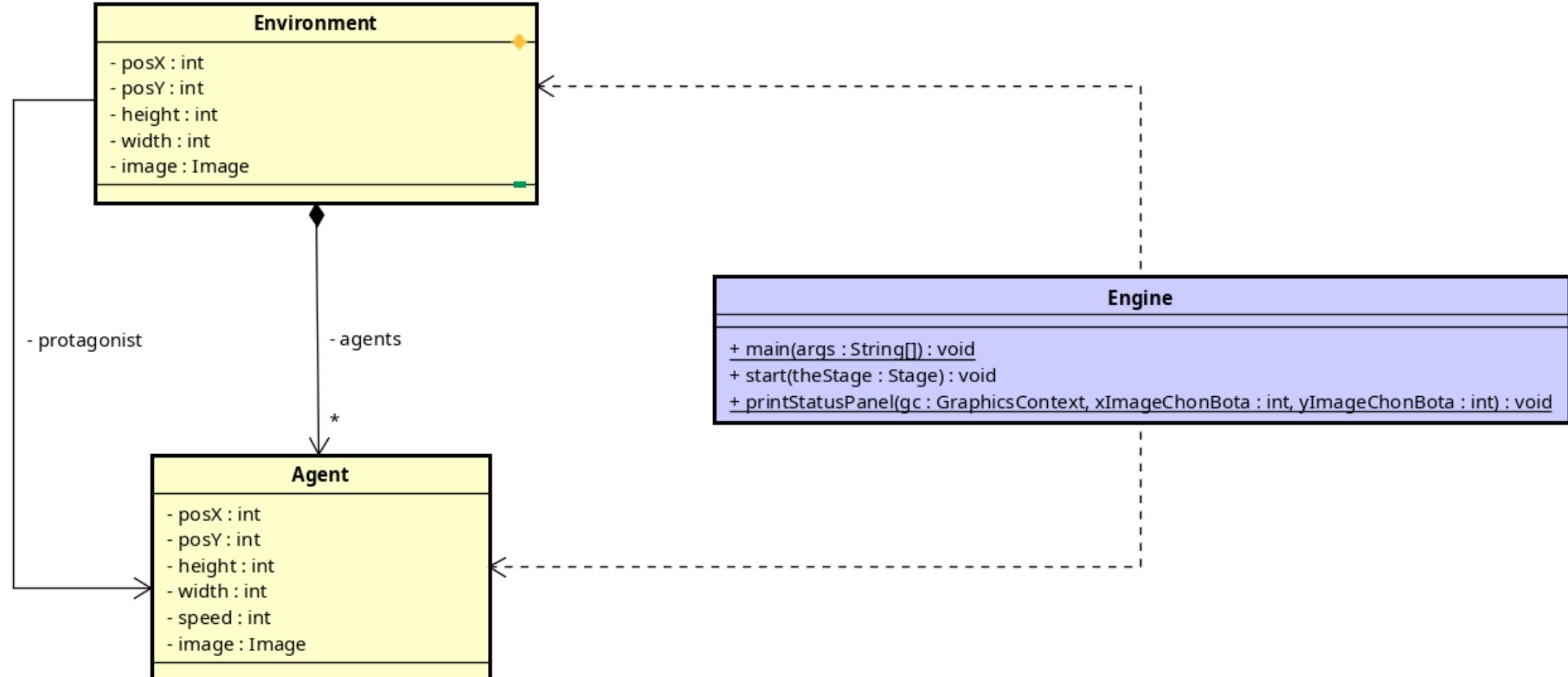
# The Class Diagram



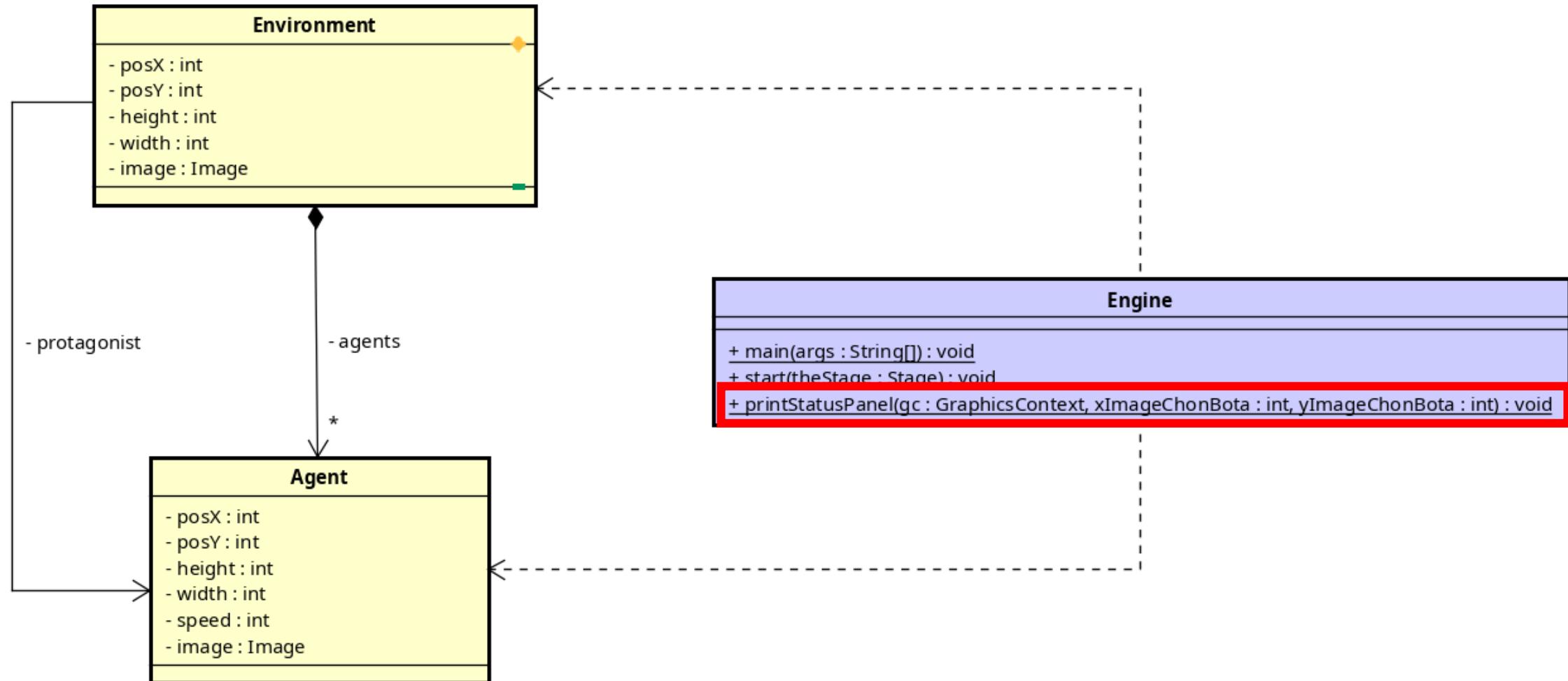
# The Class Diagram



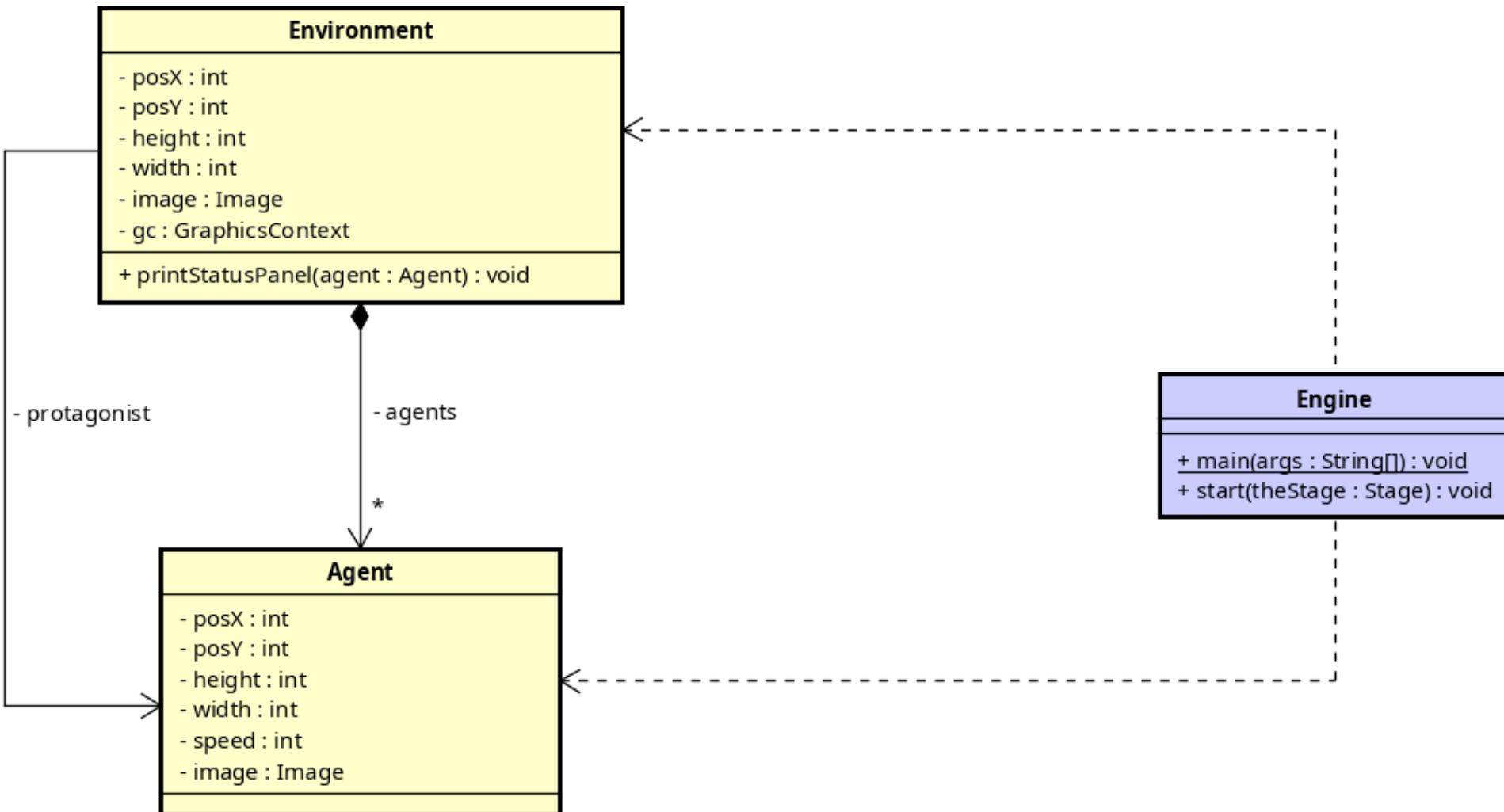
# The Class Diagram



# The Class Diagram

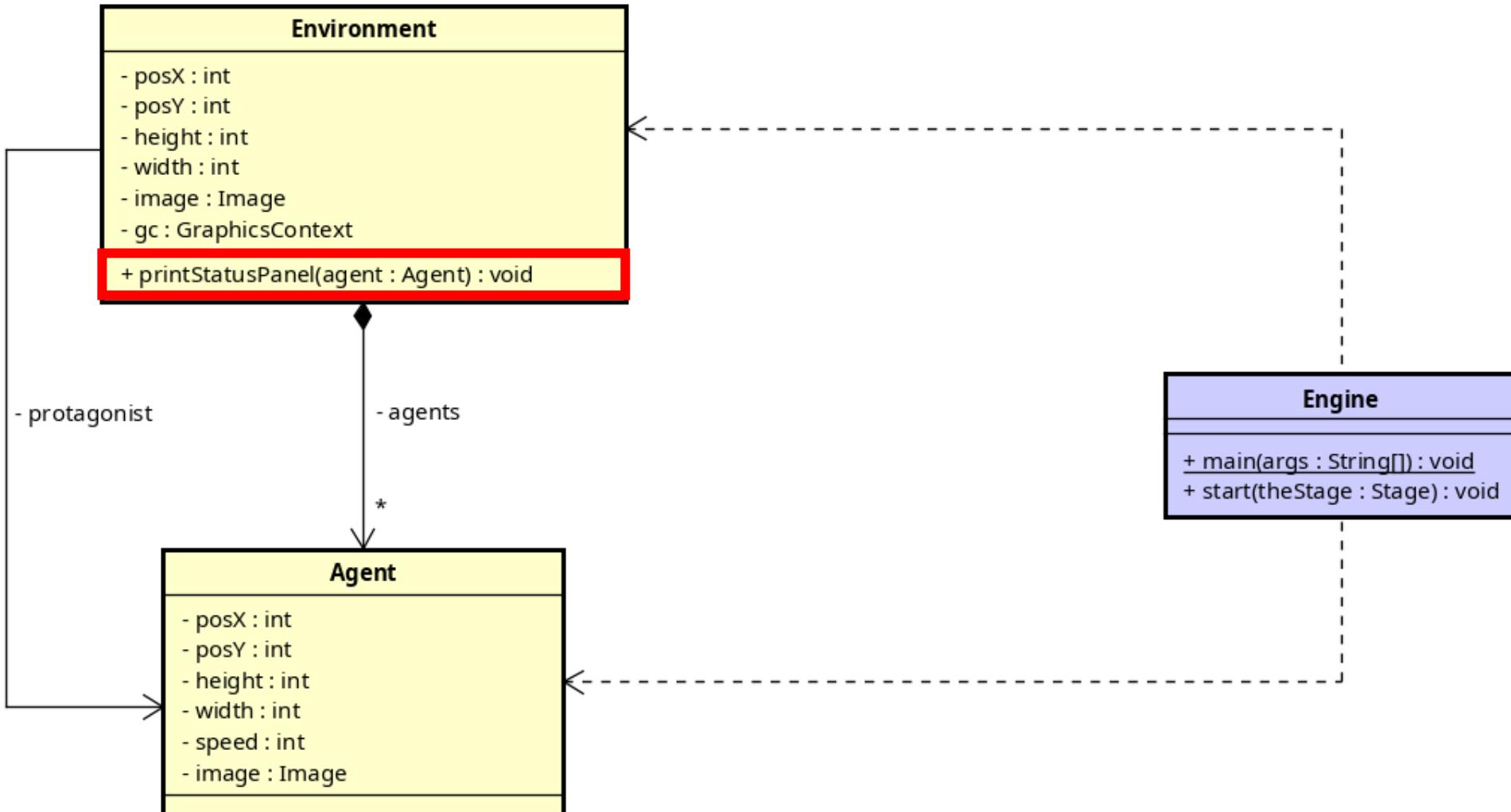


# The Class Diagram

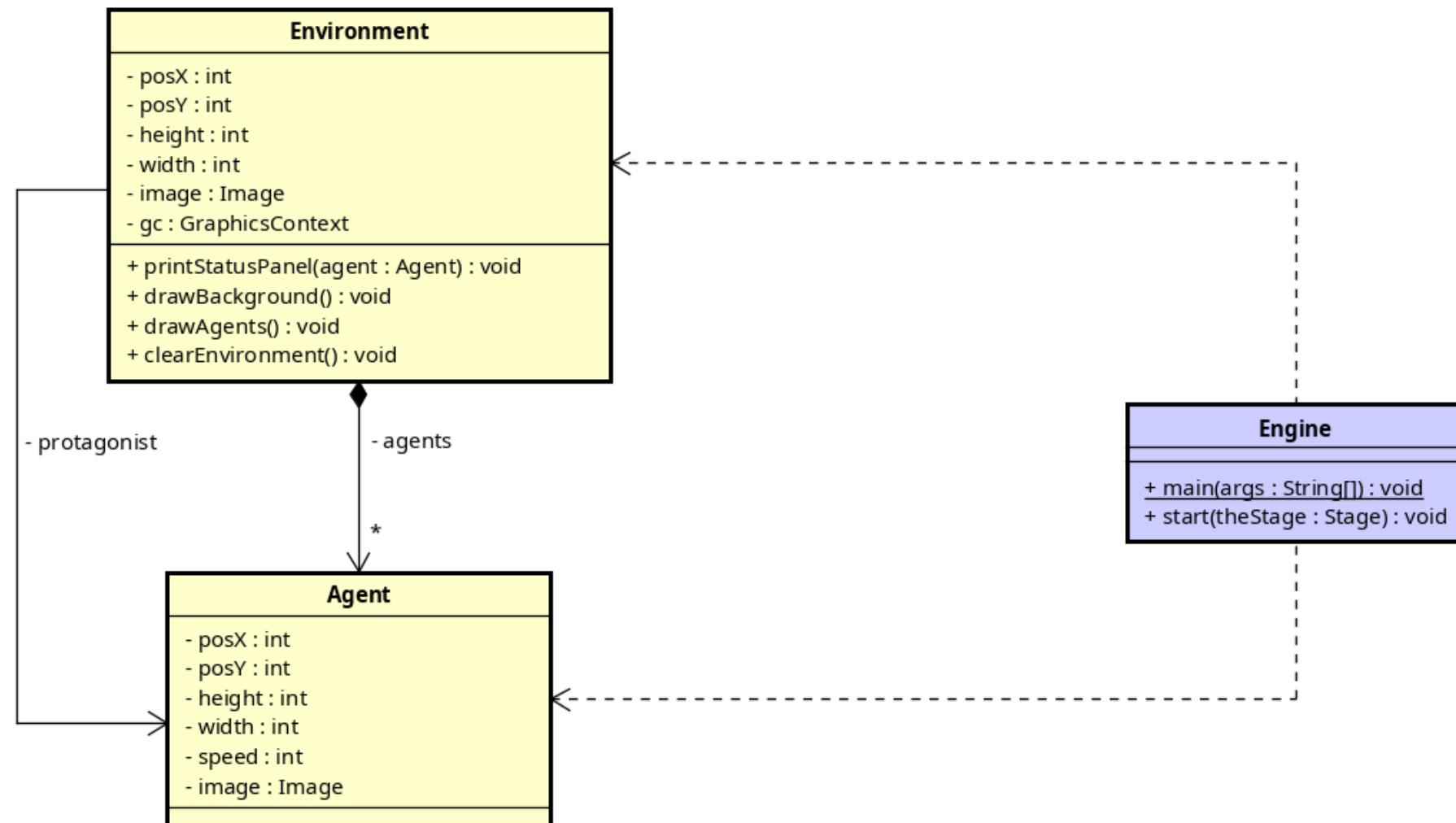


# The Class Diagram

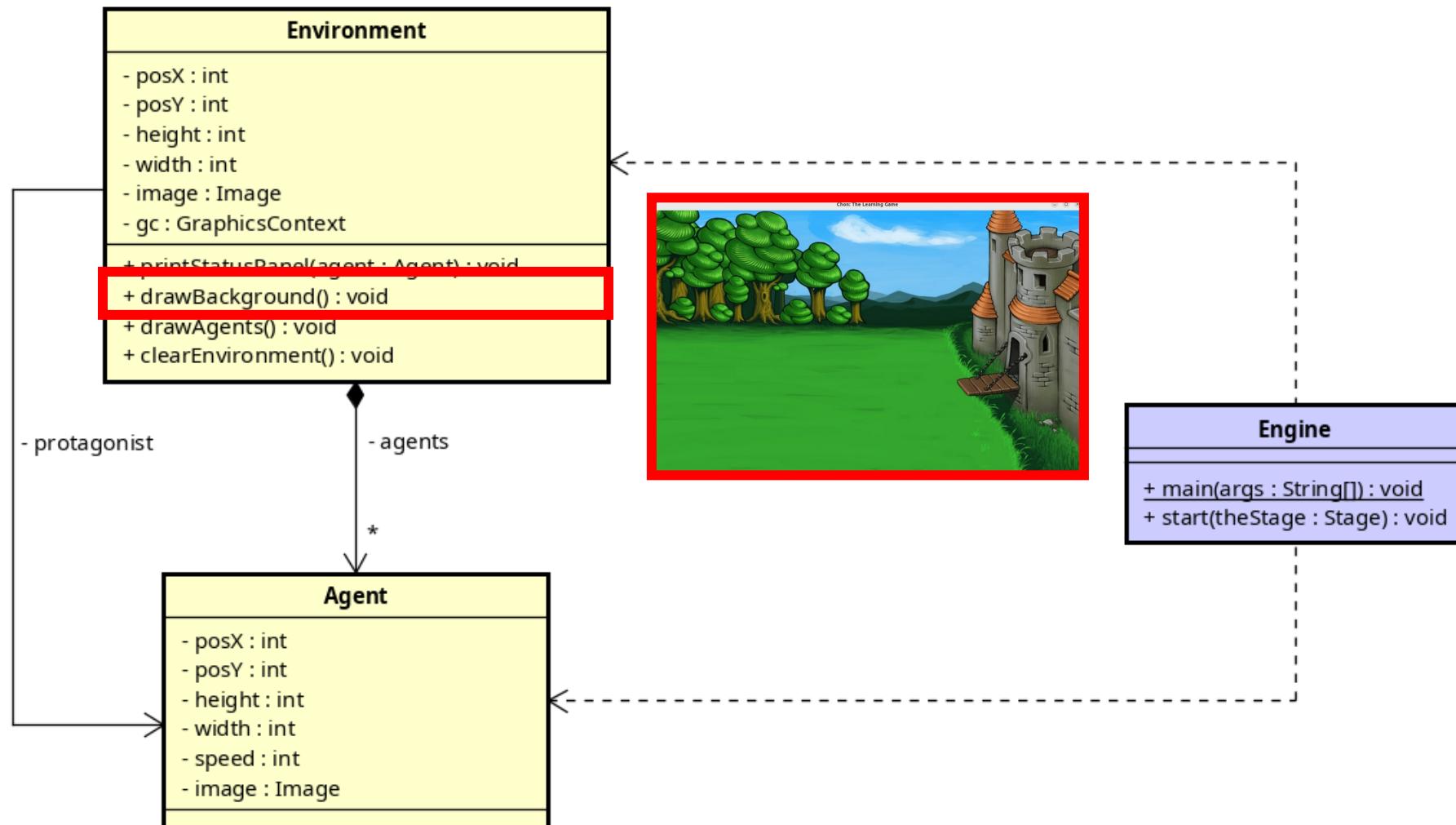
X: 400  
Y: 390



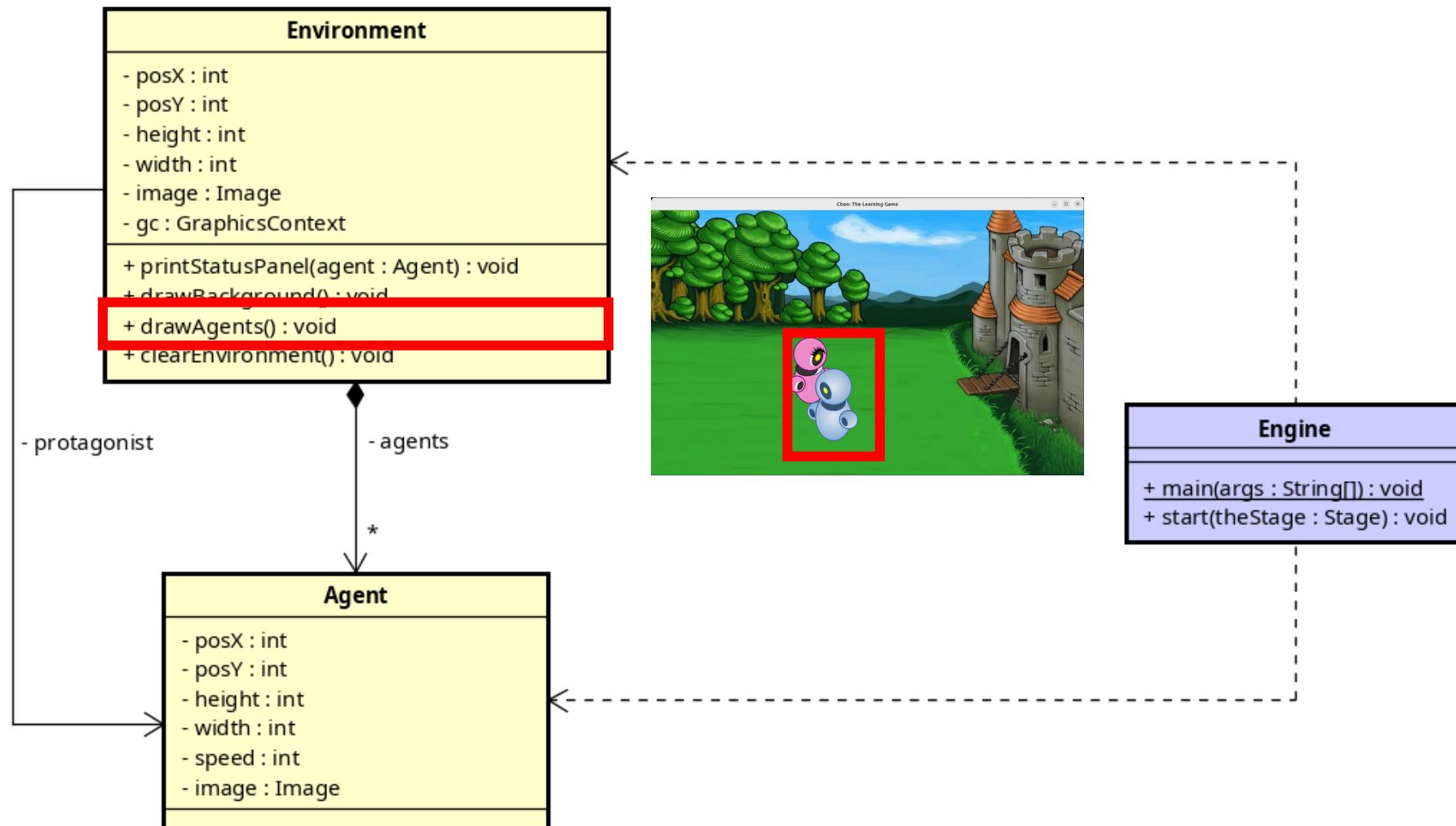
# The Class Diagram



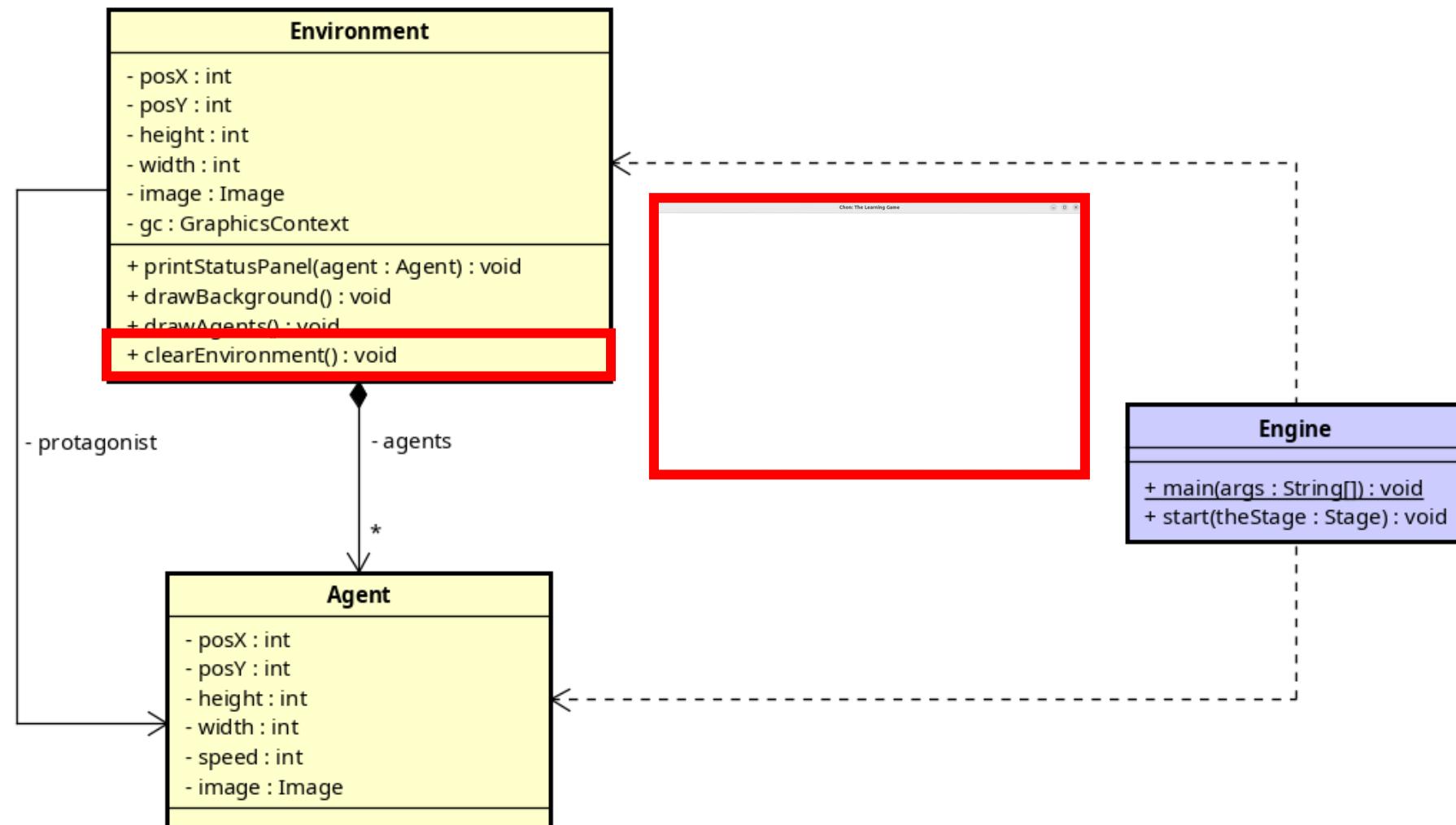
# The Class Diagram



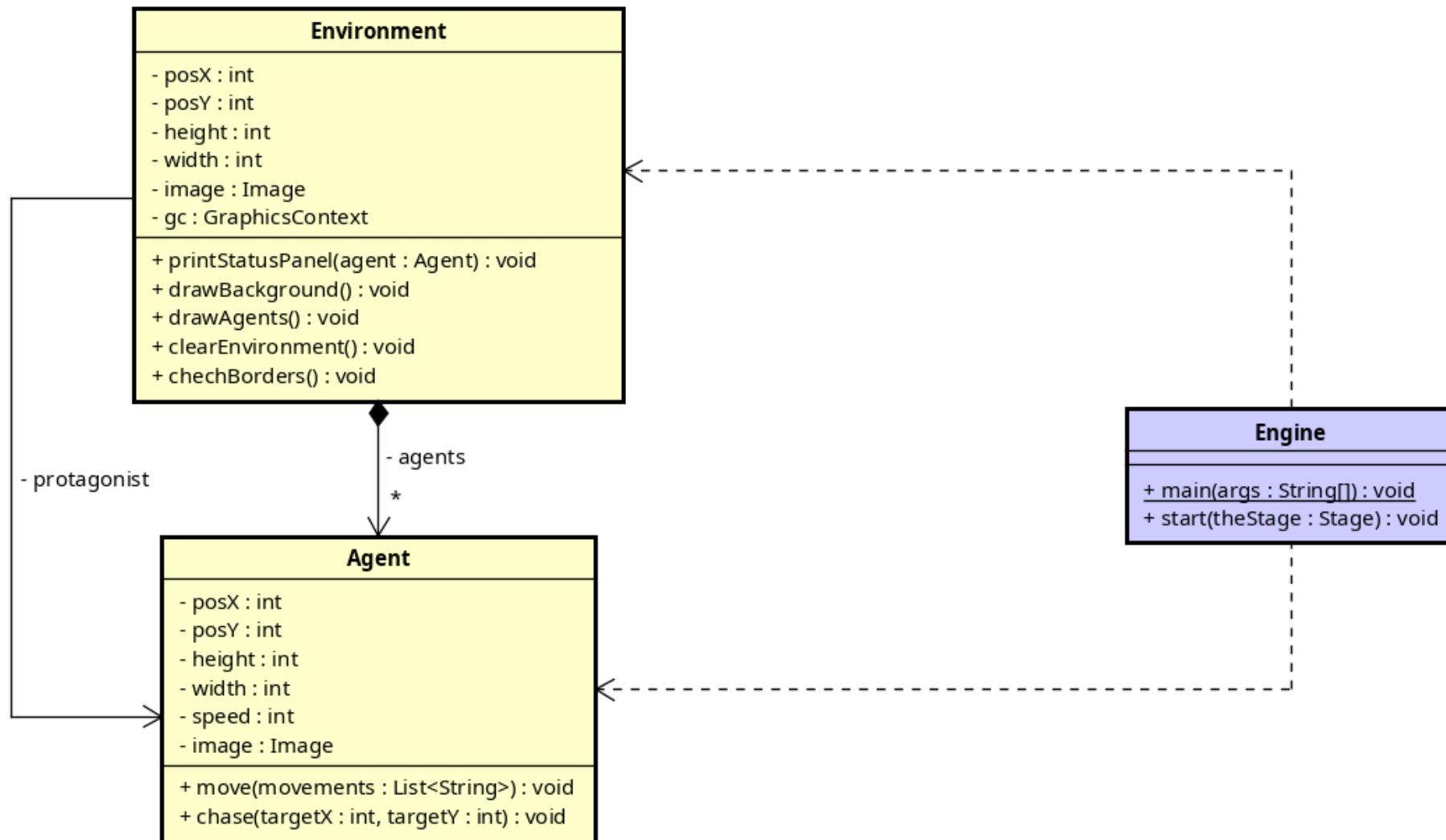
# The Class Diagram



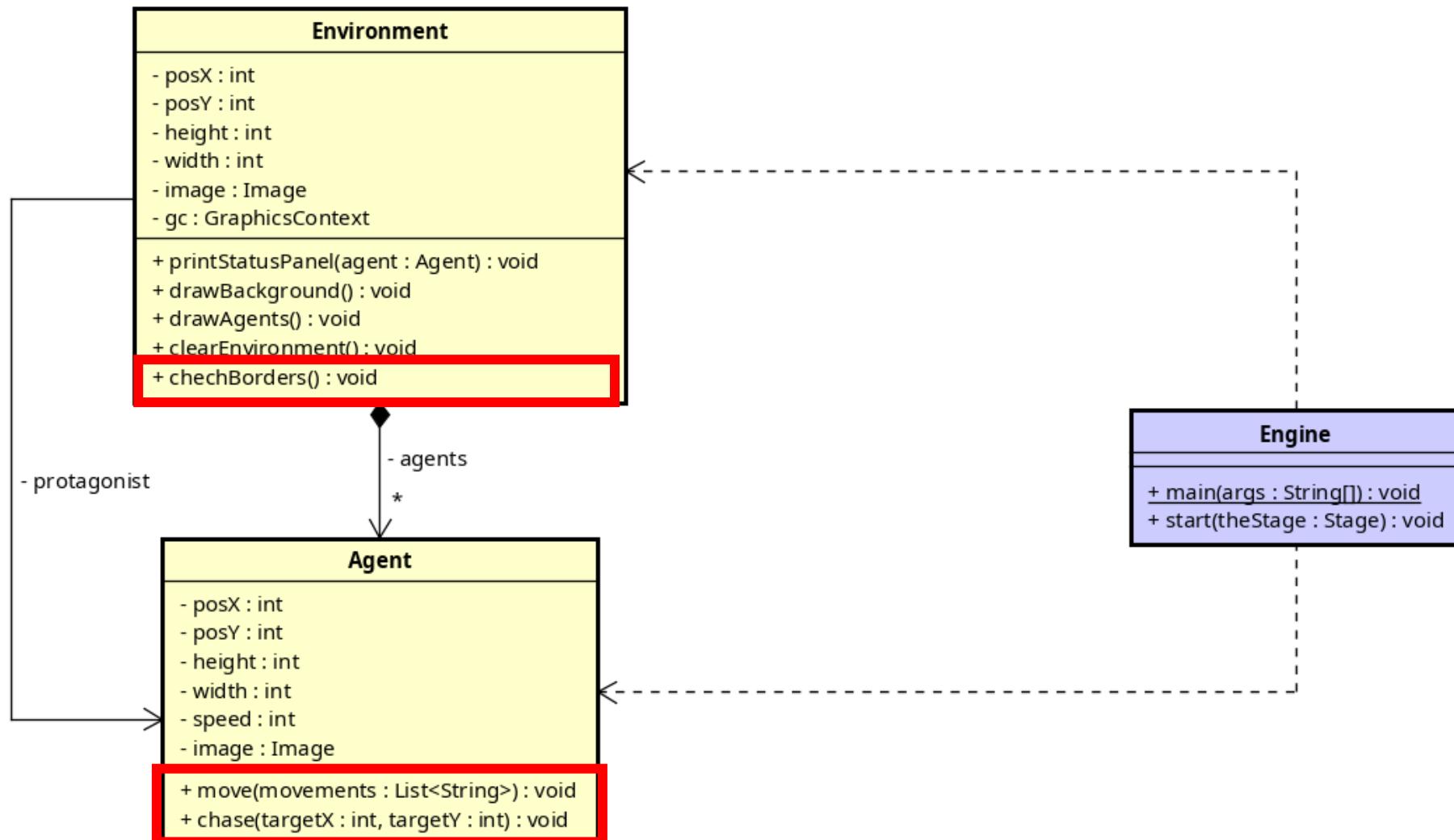
# The Class Diagram



# The Class Diagram



# The Class Diagram



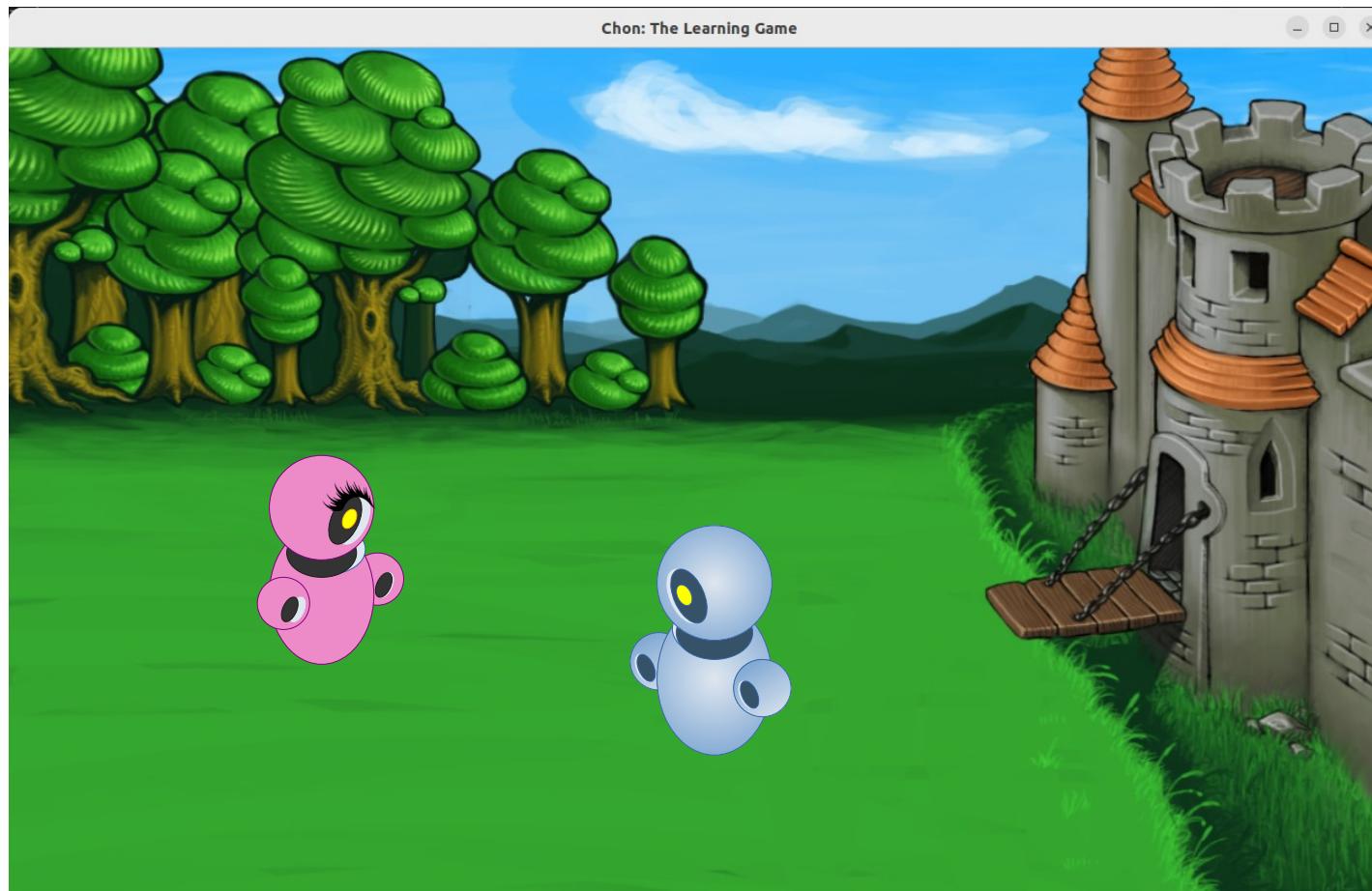
# COLLISION



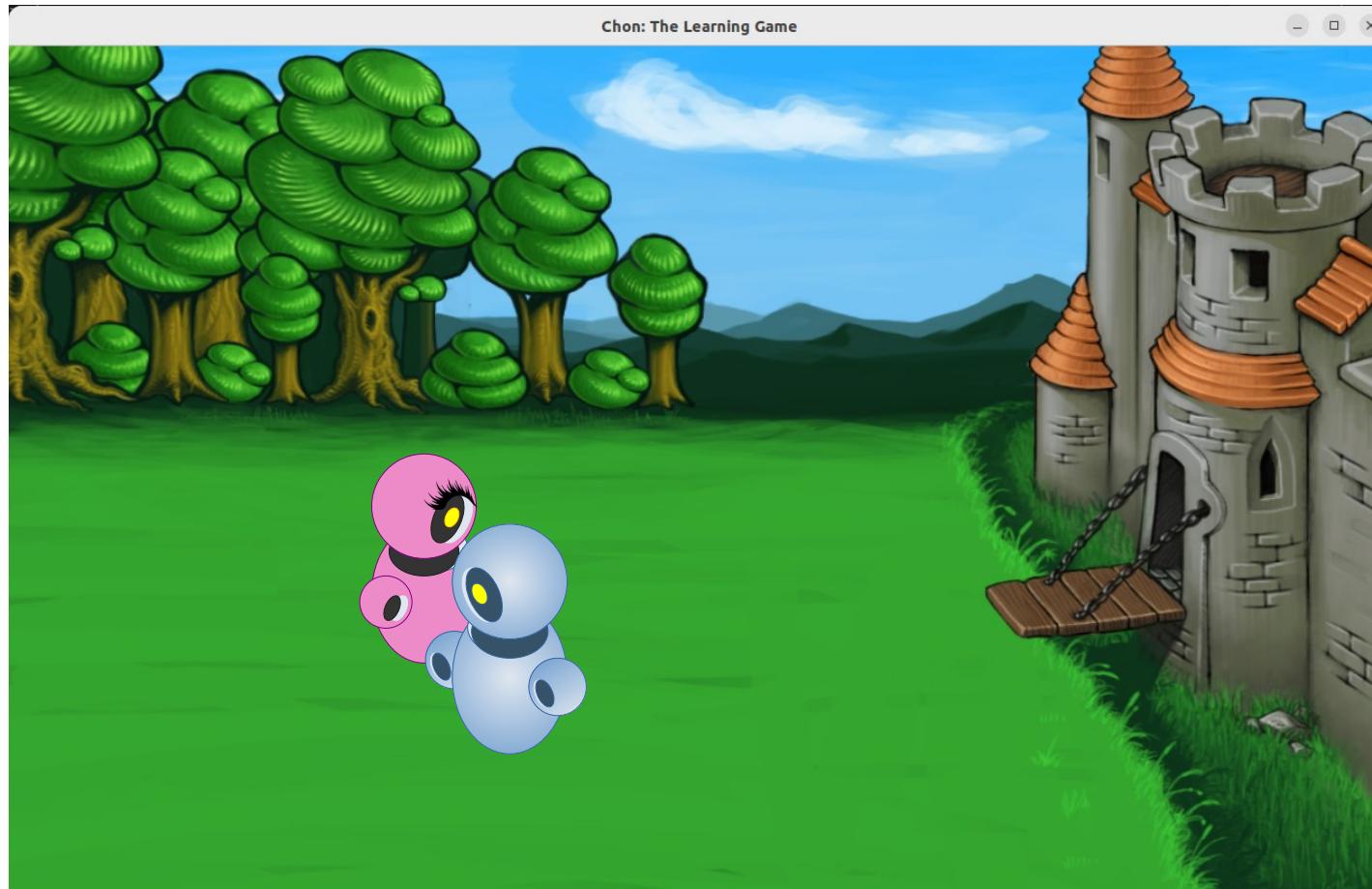
# Collision

The **collision** is an event where two or more objects in a game interact by coming into **contact** with each other.

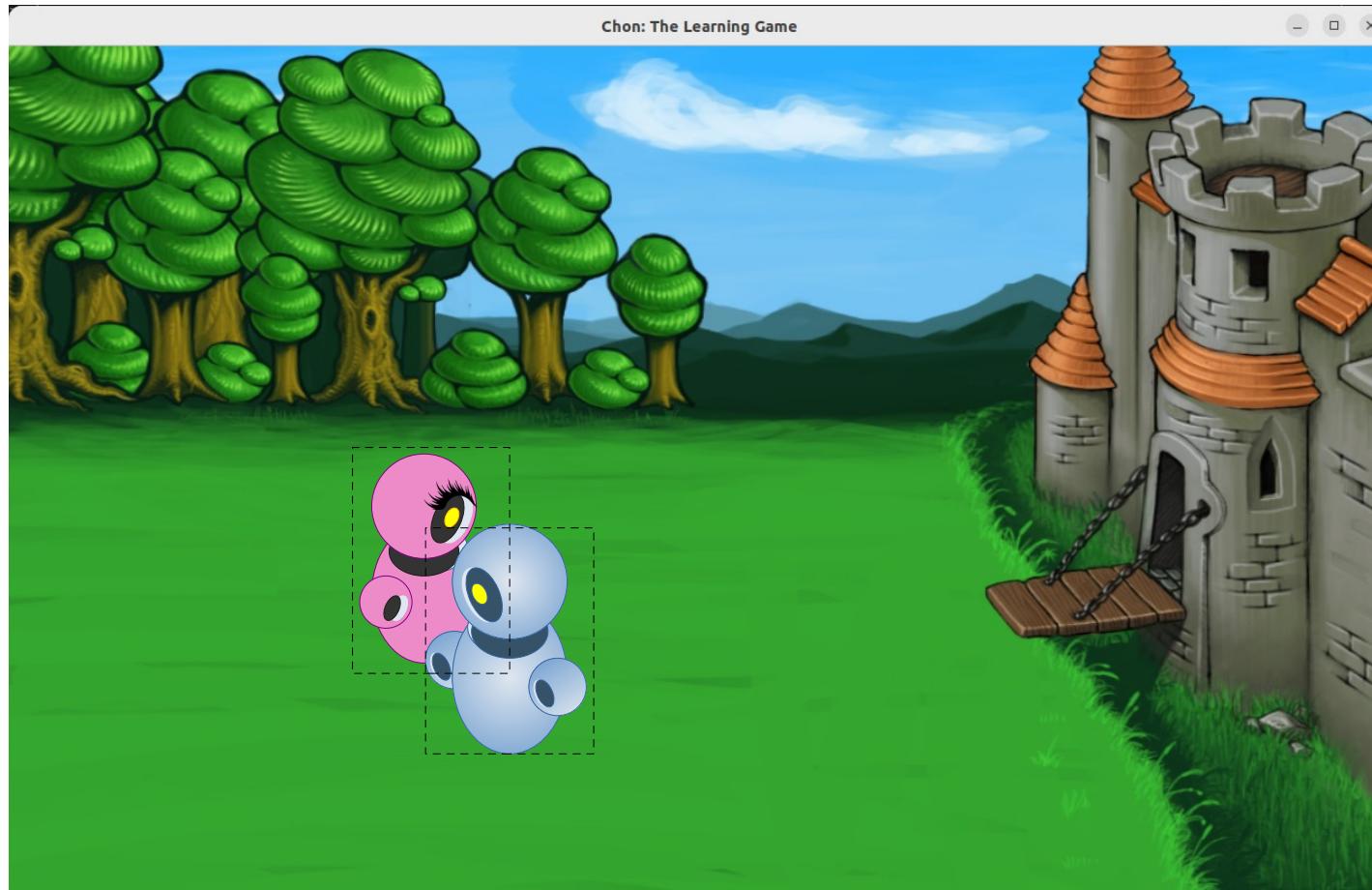
# Collision



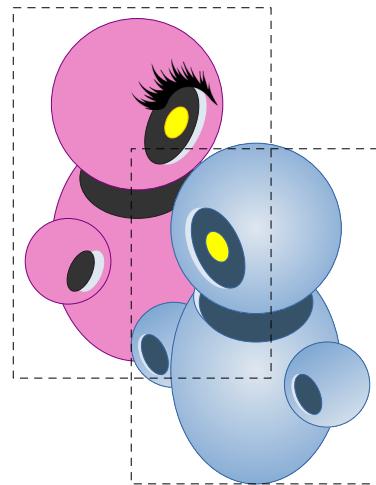
# Collision



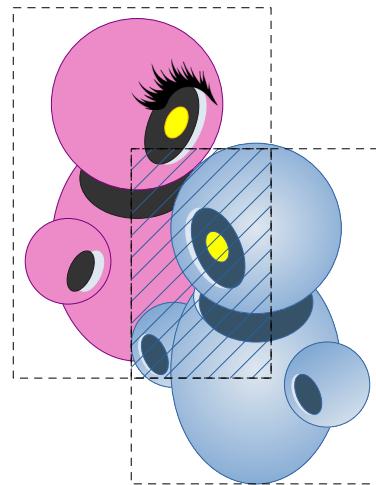
# Collision



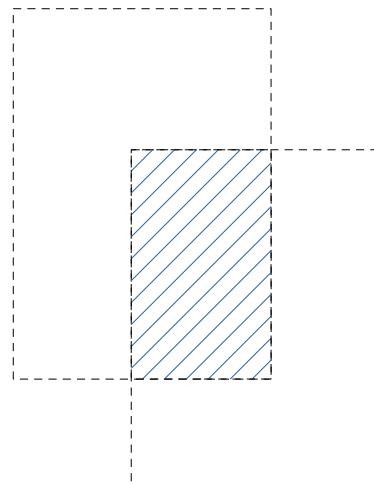
# Collision



# Collision



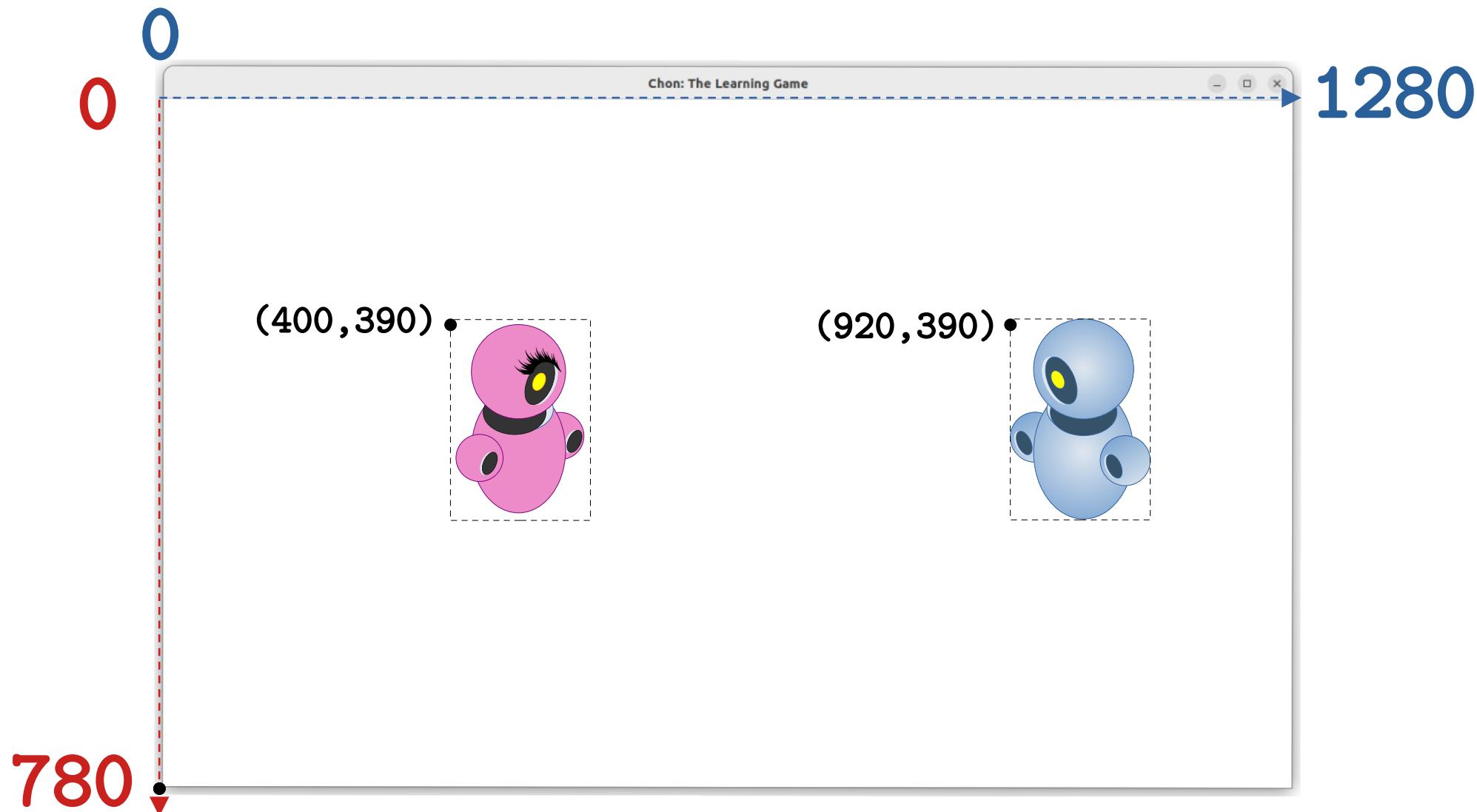
# Collision



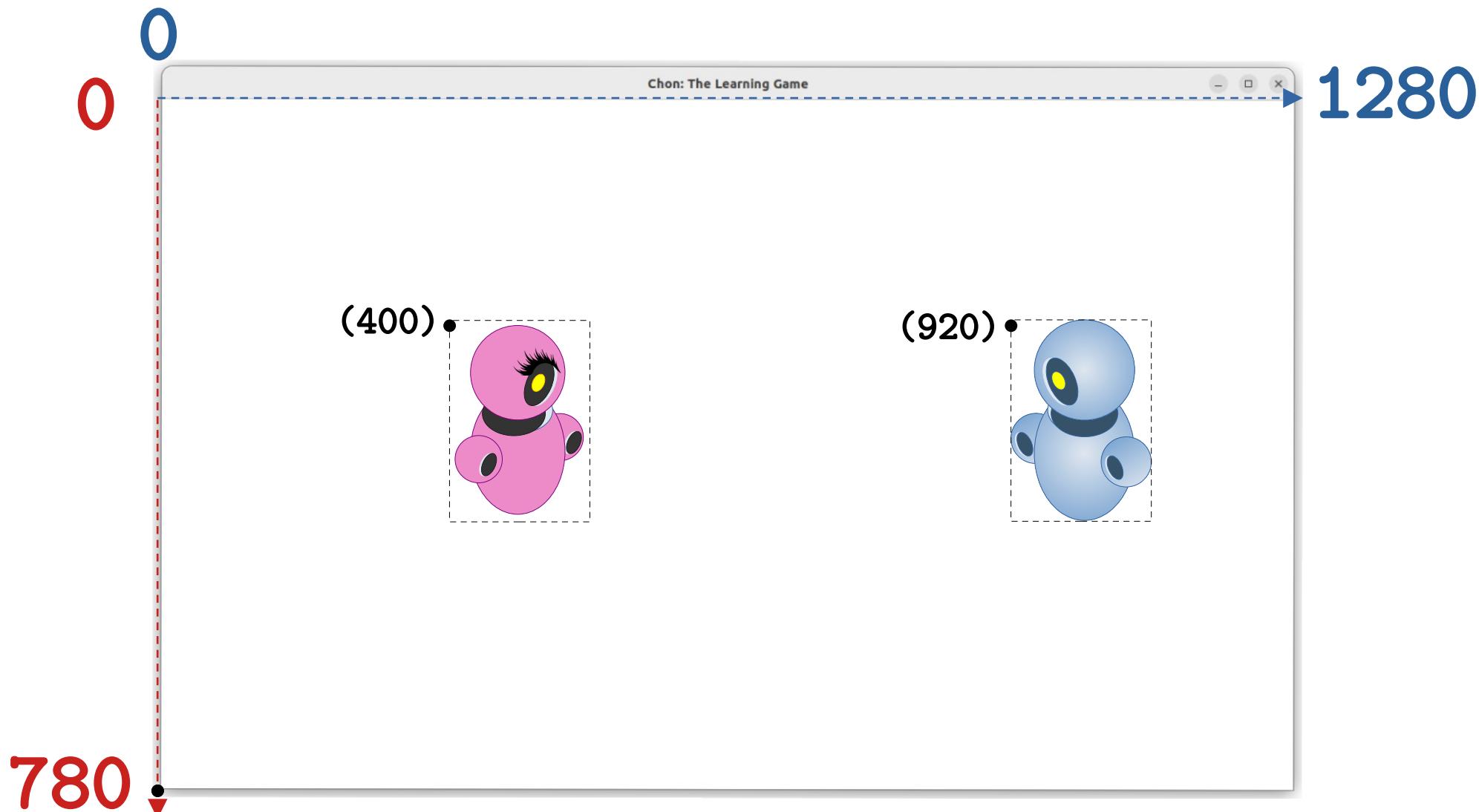
# Axis X: Condition 1

The **protagonist** must not **outrun** the other agents.

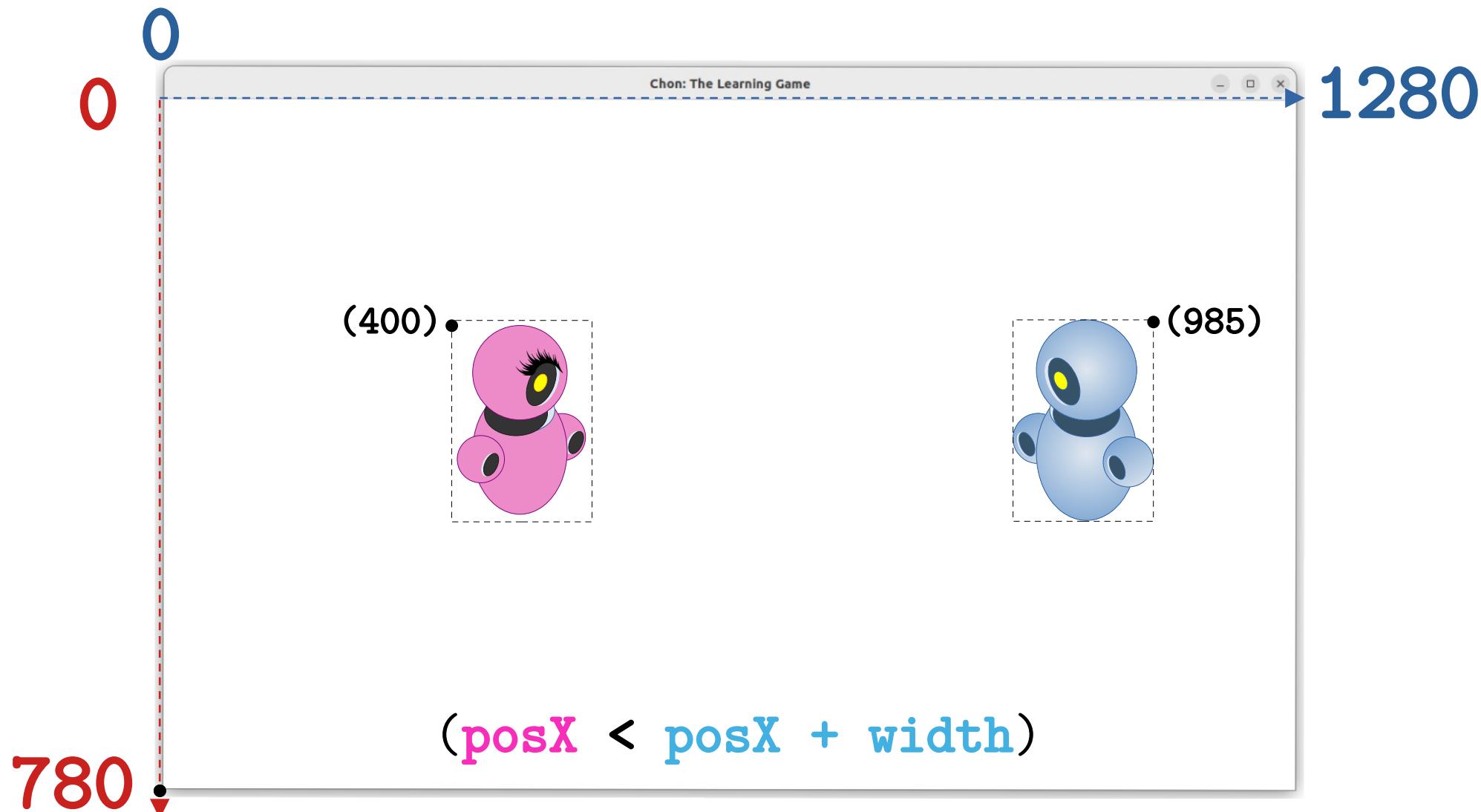
# Axis X: Condition 1



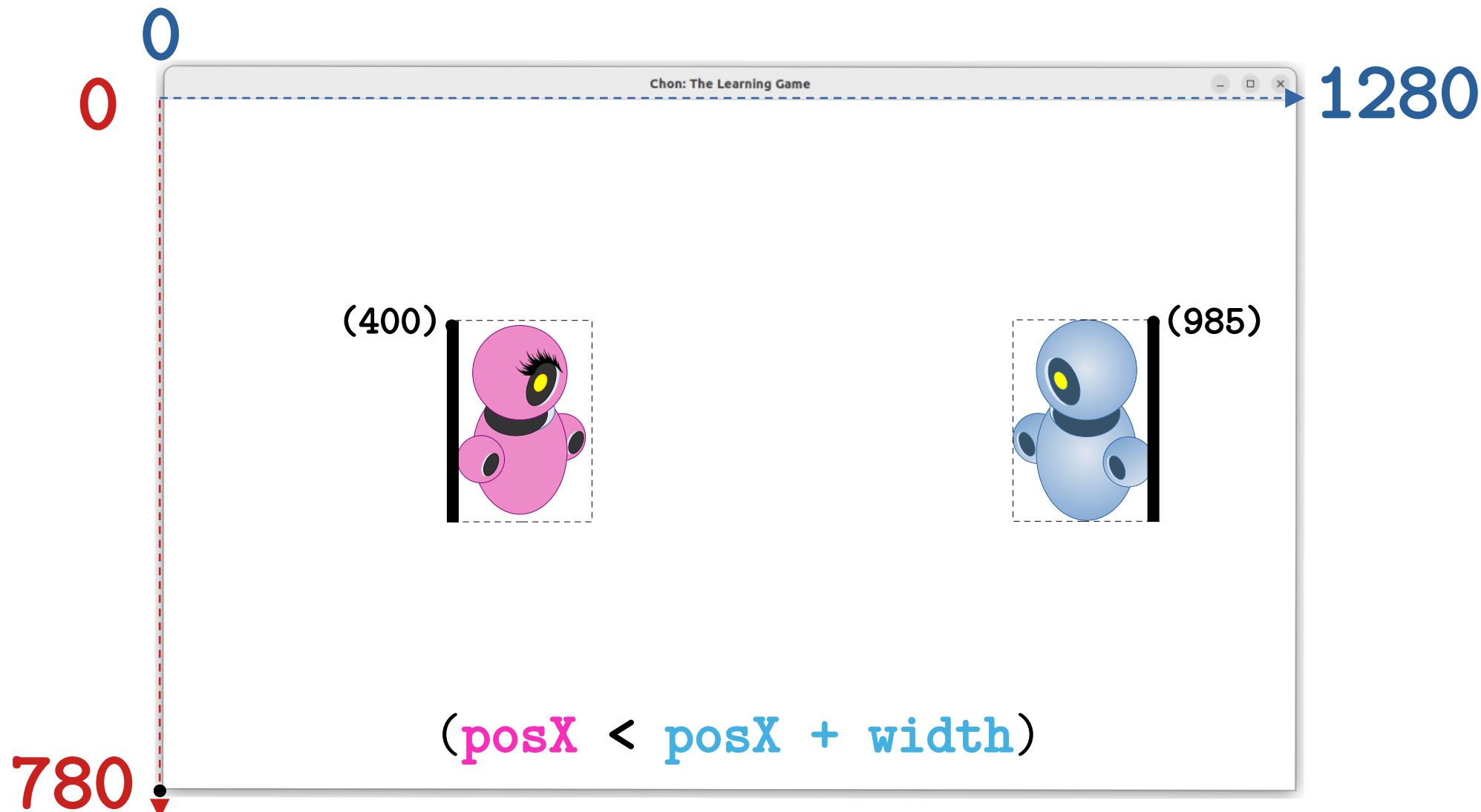
# Axis X: Condition 1



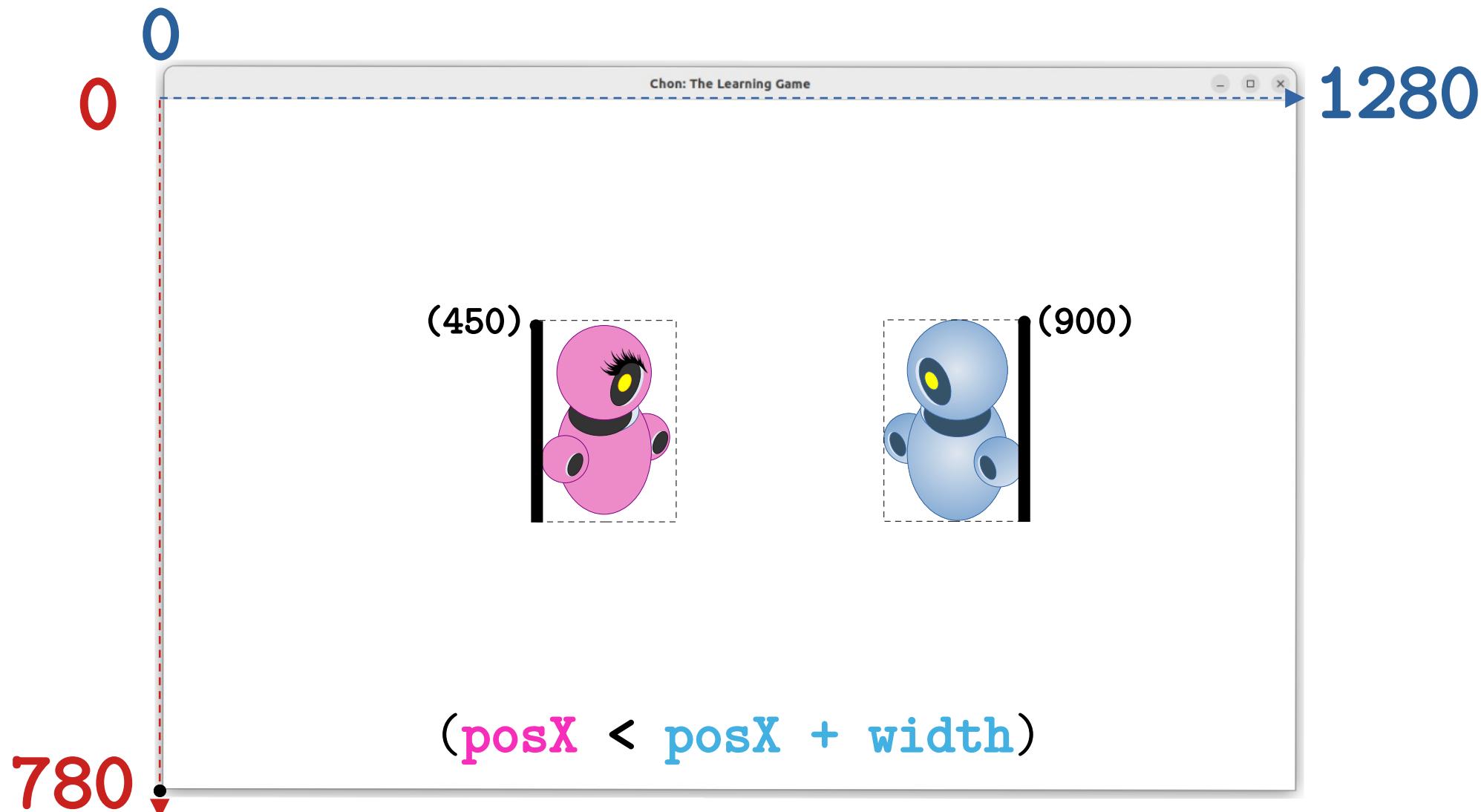
# Axis X: Condition 1



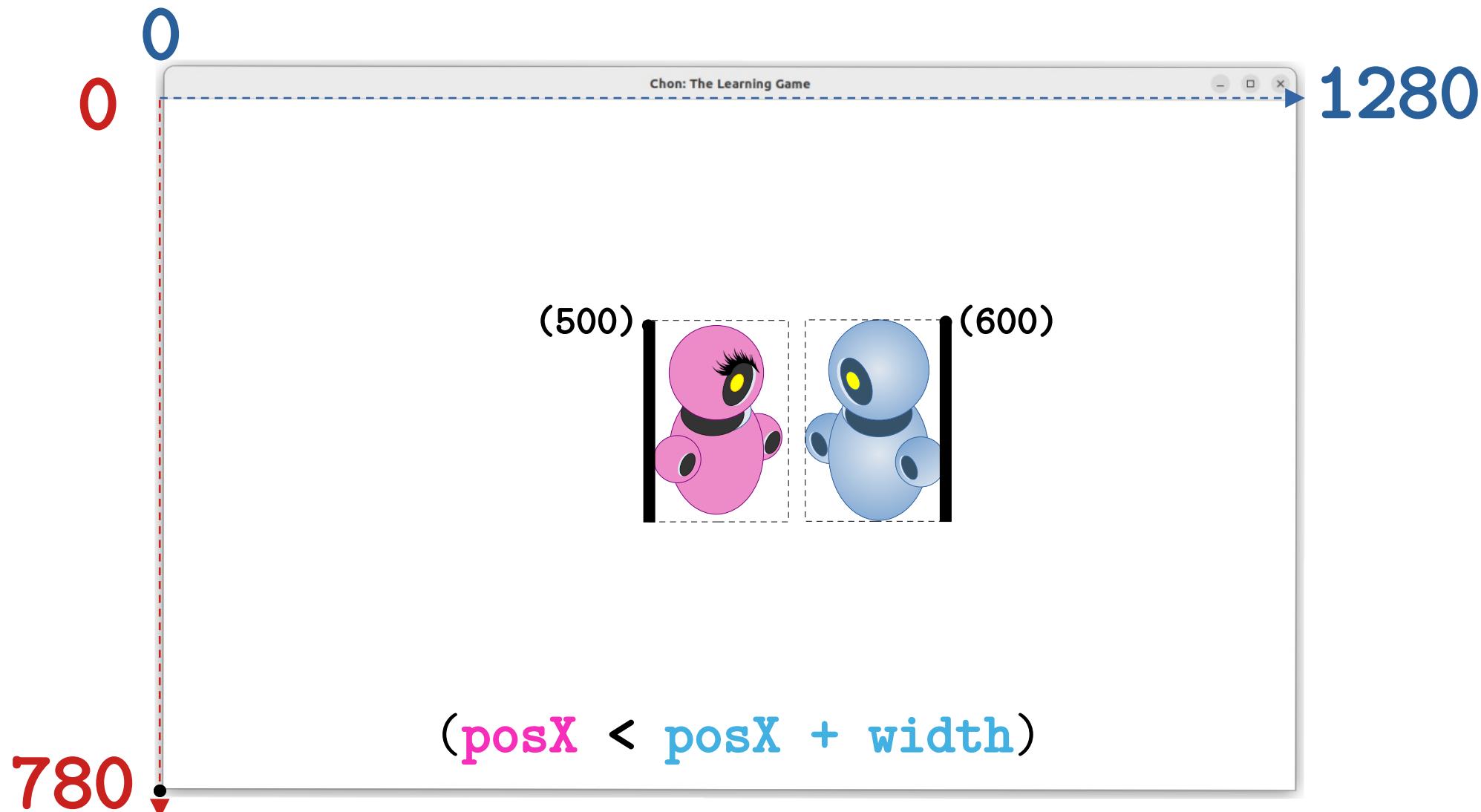
# Axis X: Condition 1



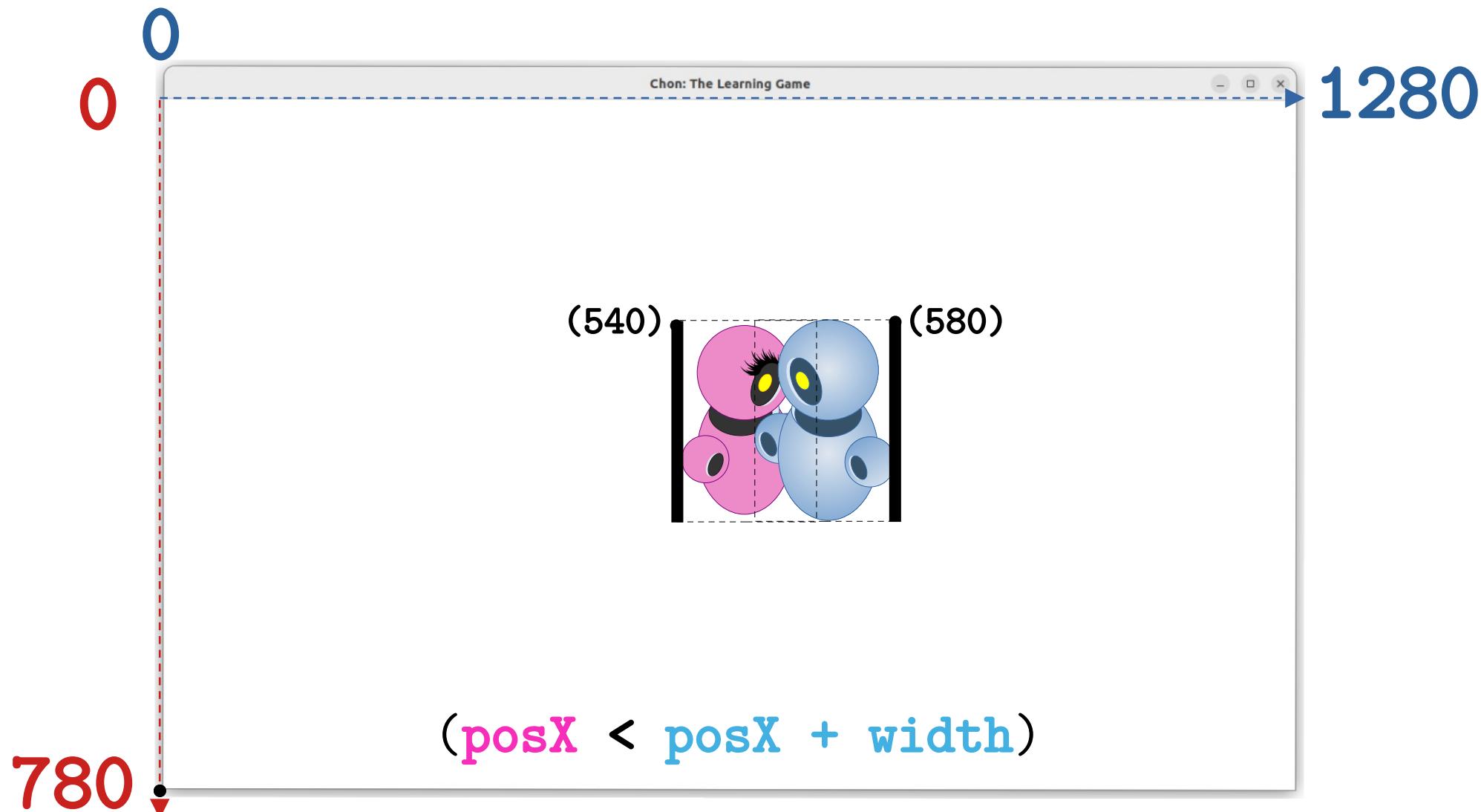
# Axis X: Condition 1



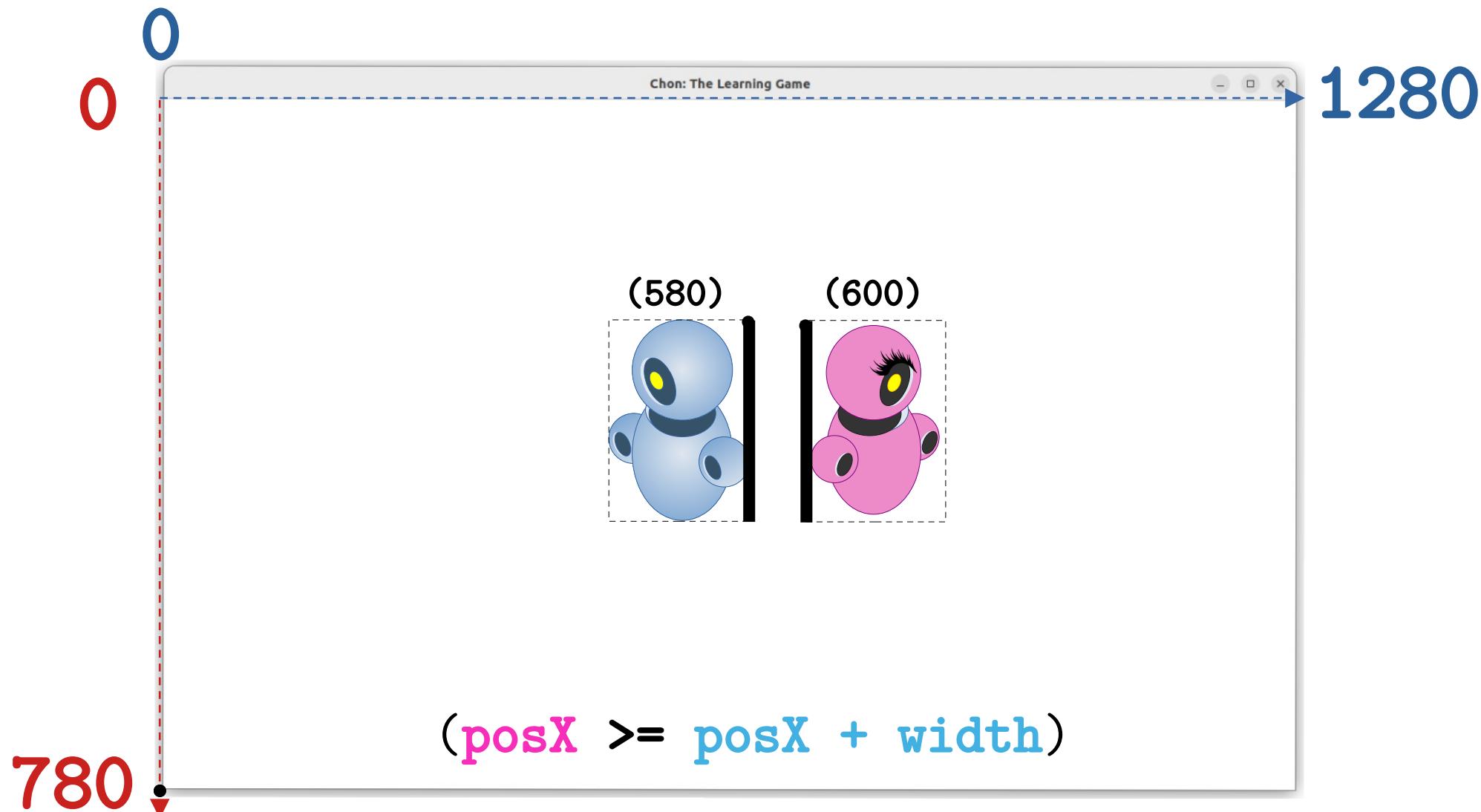
# Axis X: Condition 1



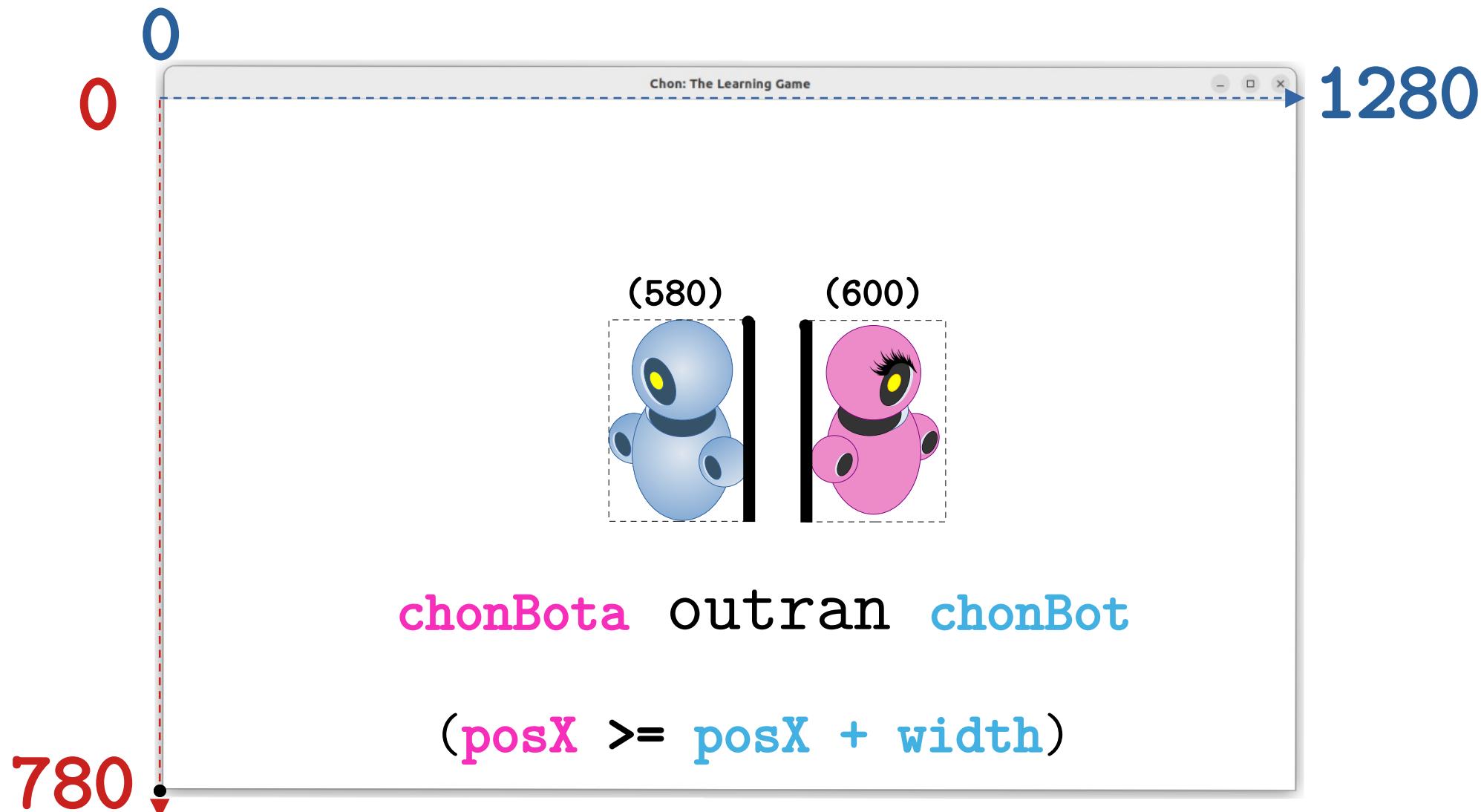
# Axis X: Condition 1



# Axis X: Condition 1



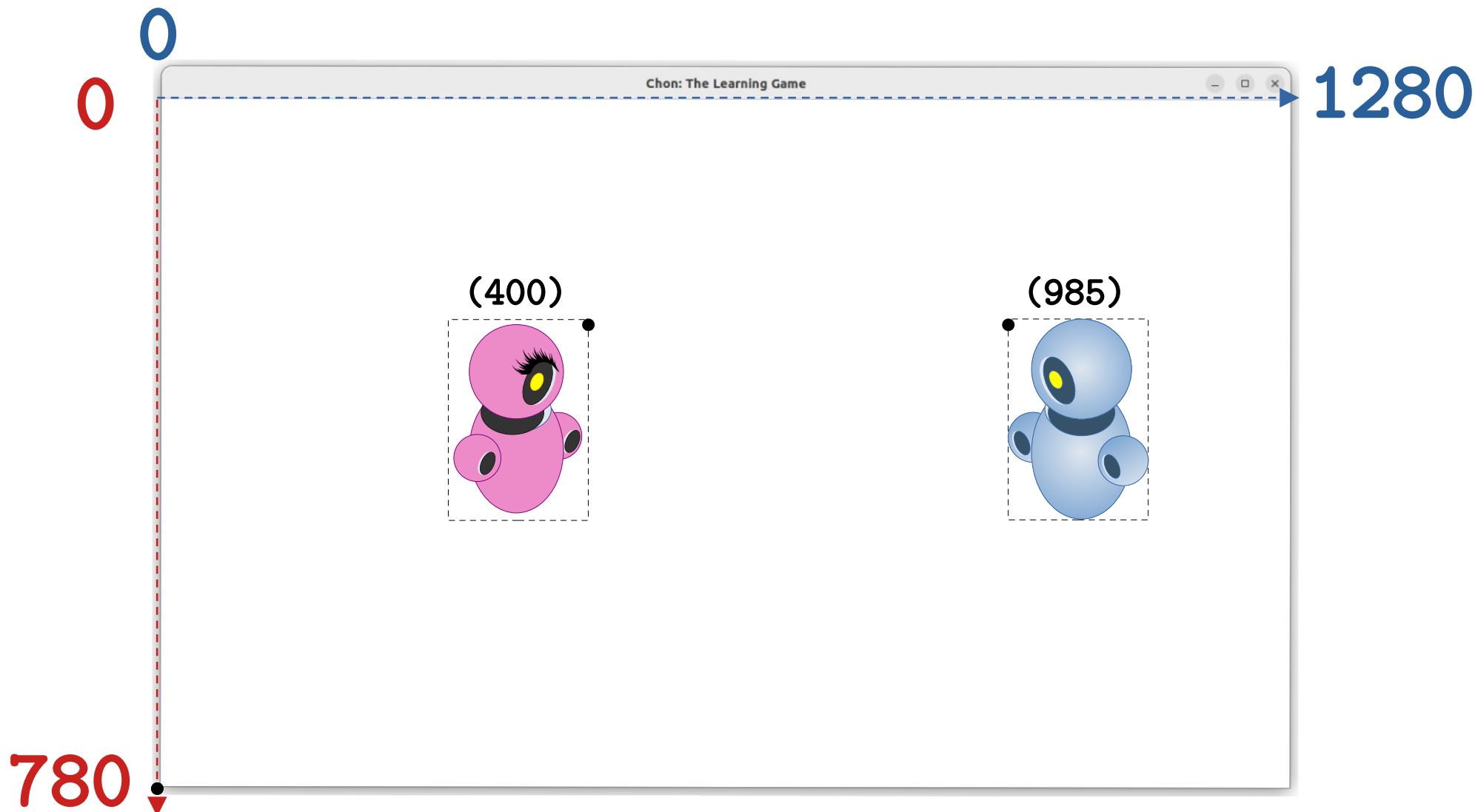
# Axis X: Condition 1



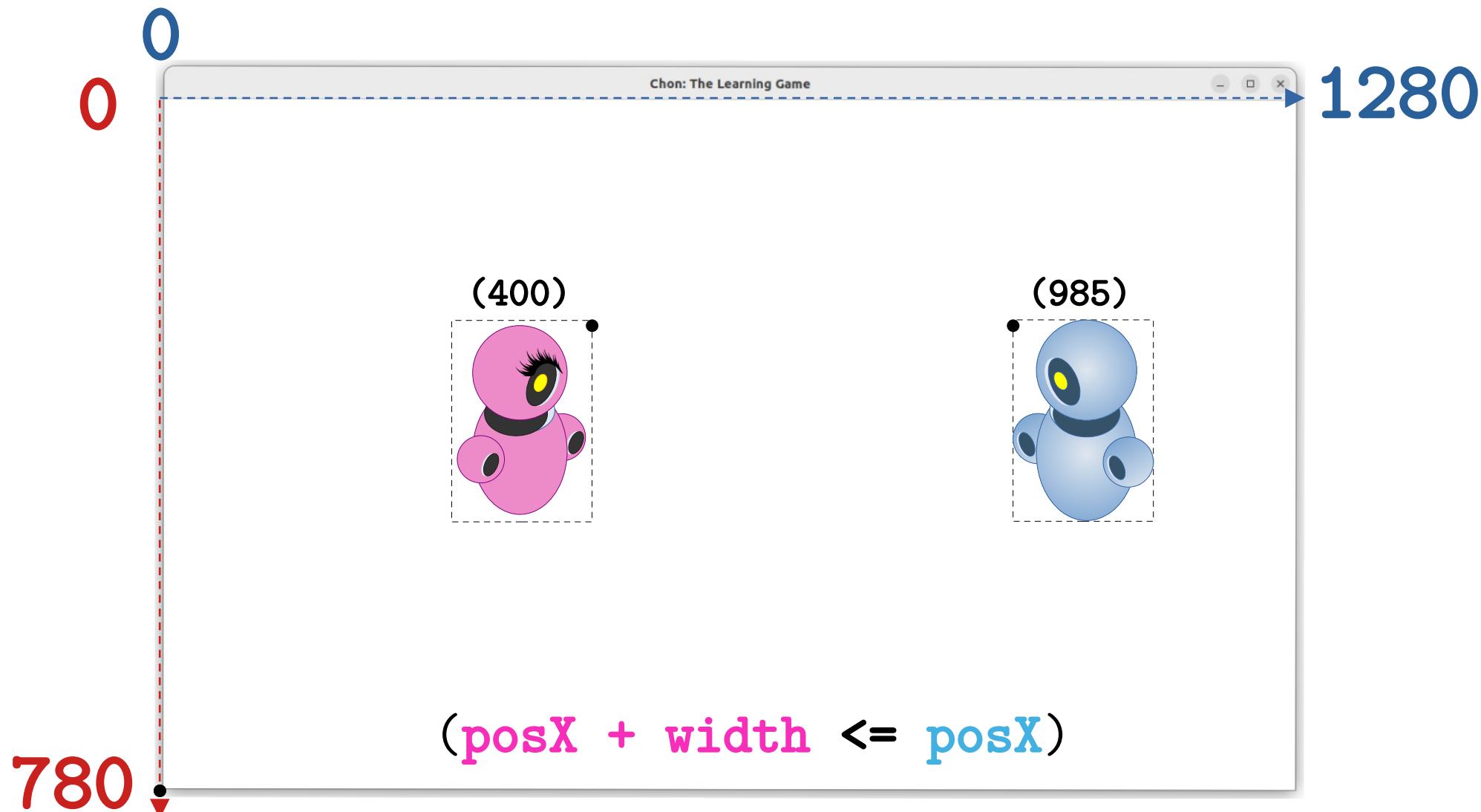
## Axis X: Condition 2

The **protagonist** must **cross** the other agents.

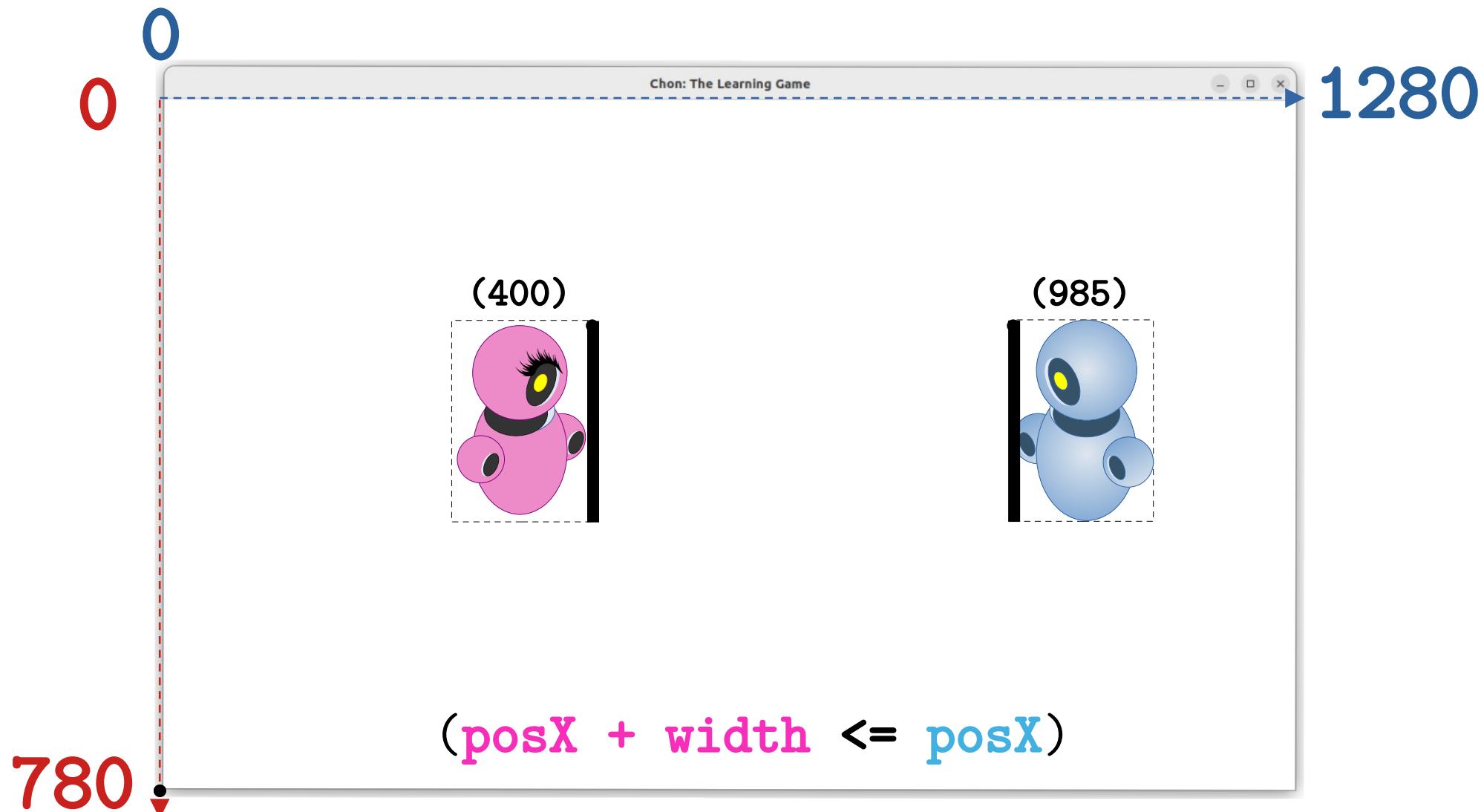
# Axis X: Condition 2



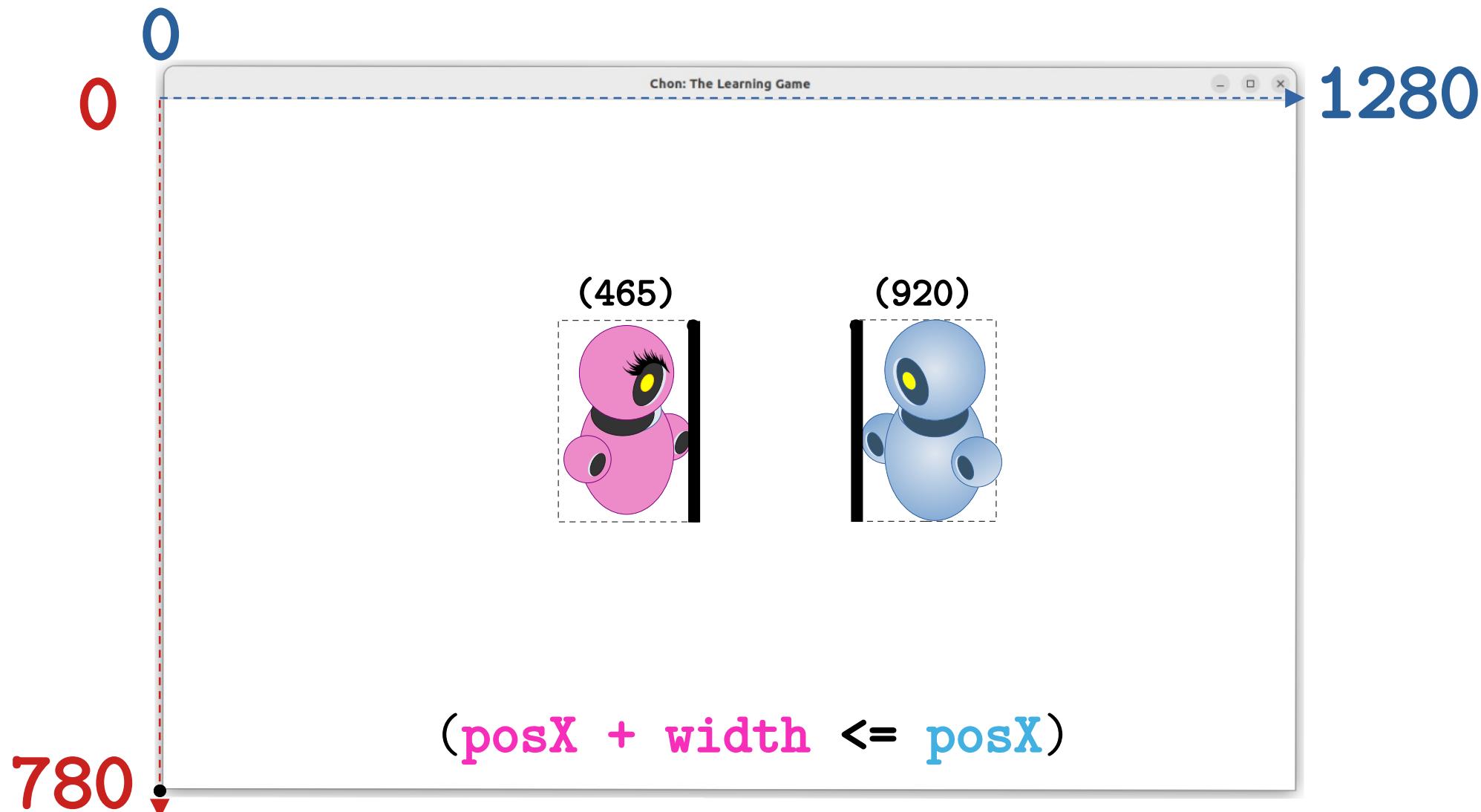
# Axis X: Condition 2



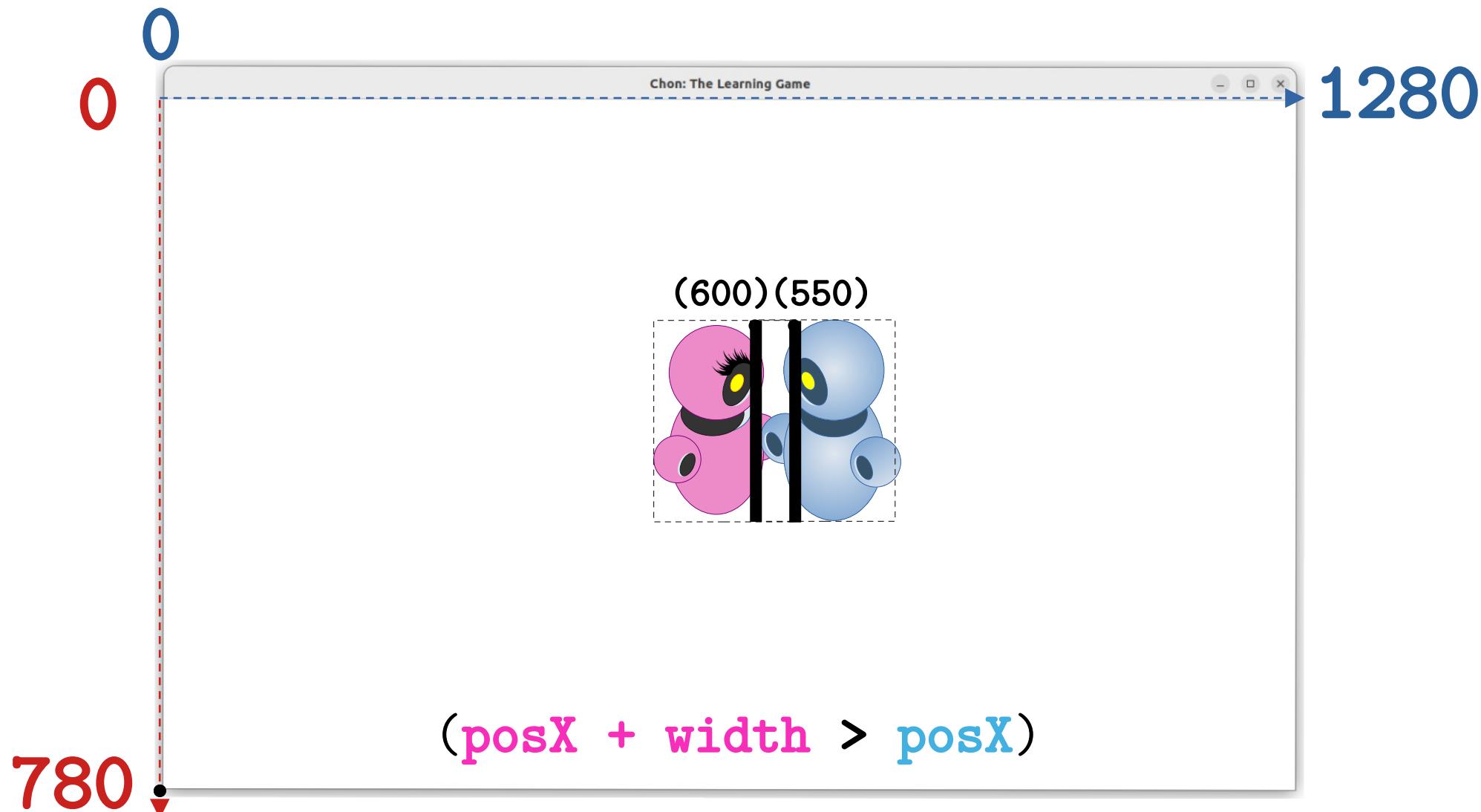
# Axis X: Condition 2



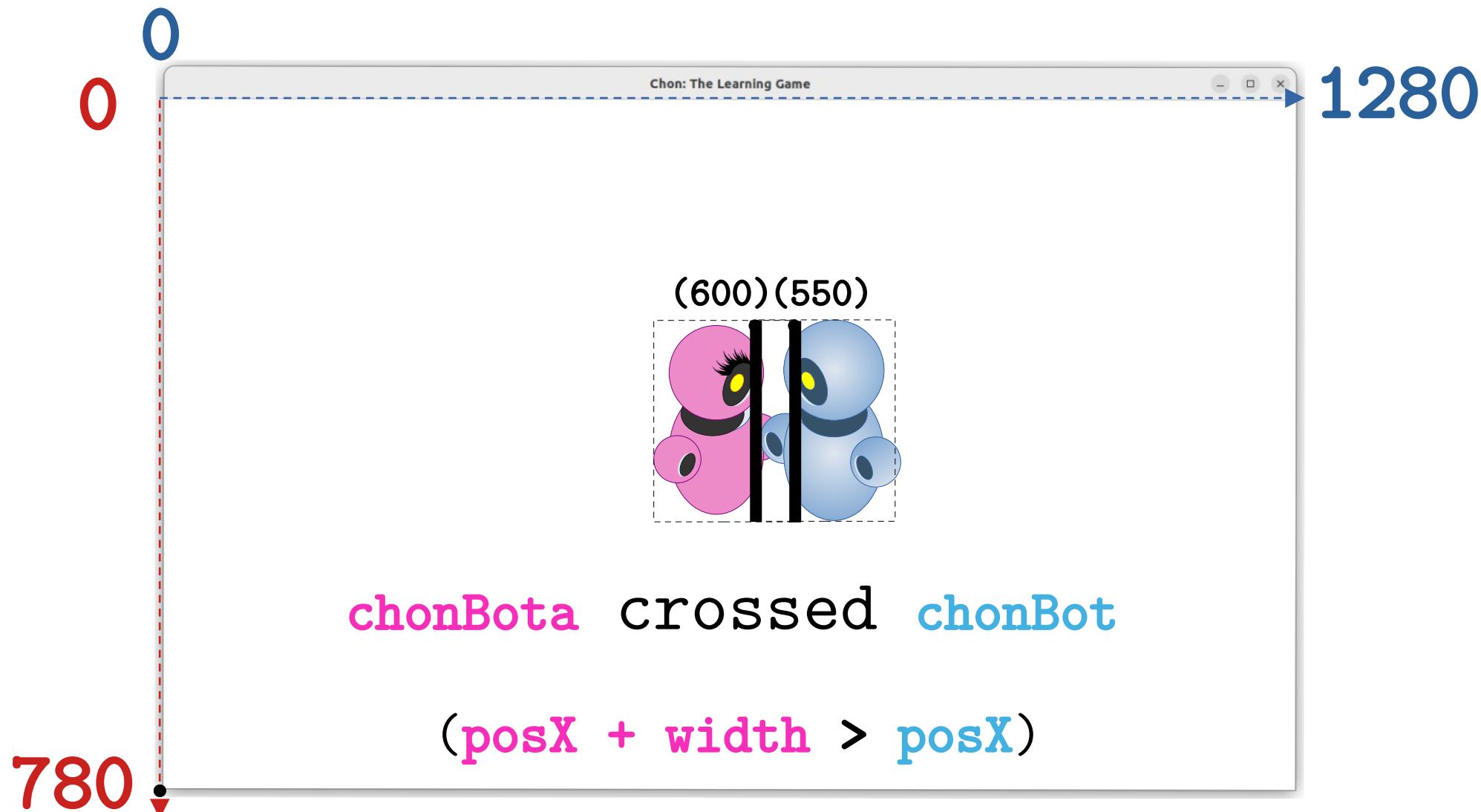
# Axis X: Condition 2



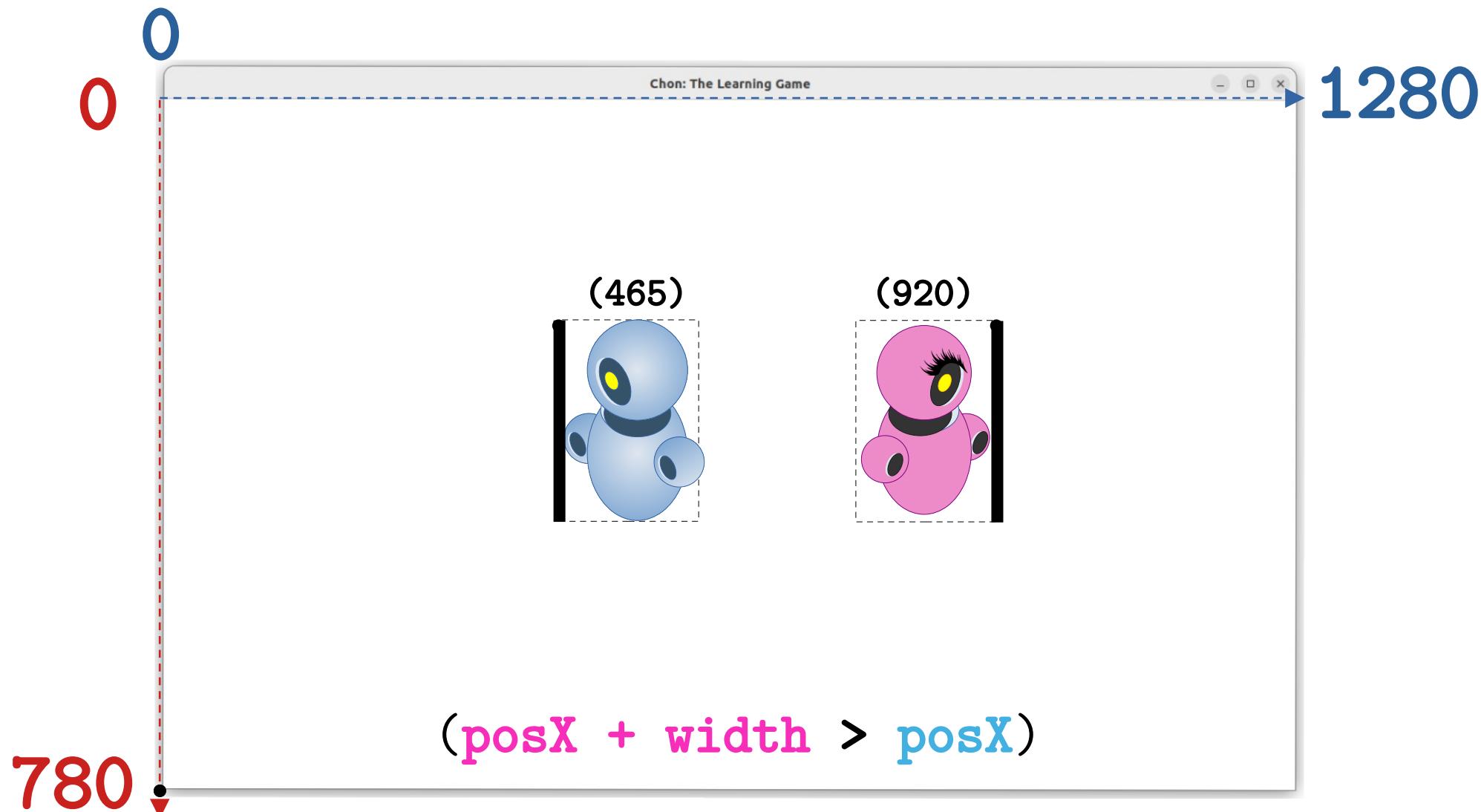
# Axis X: Condition 2



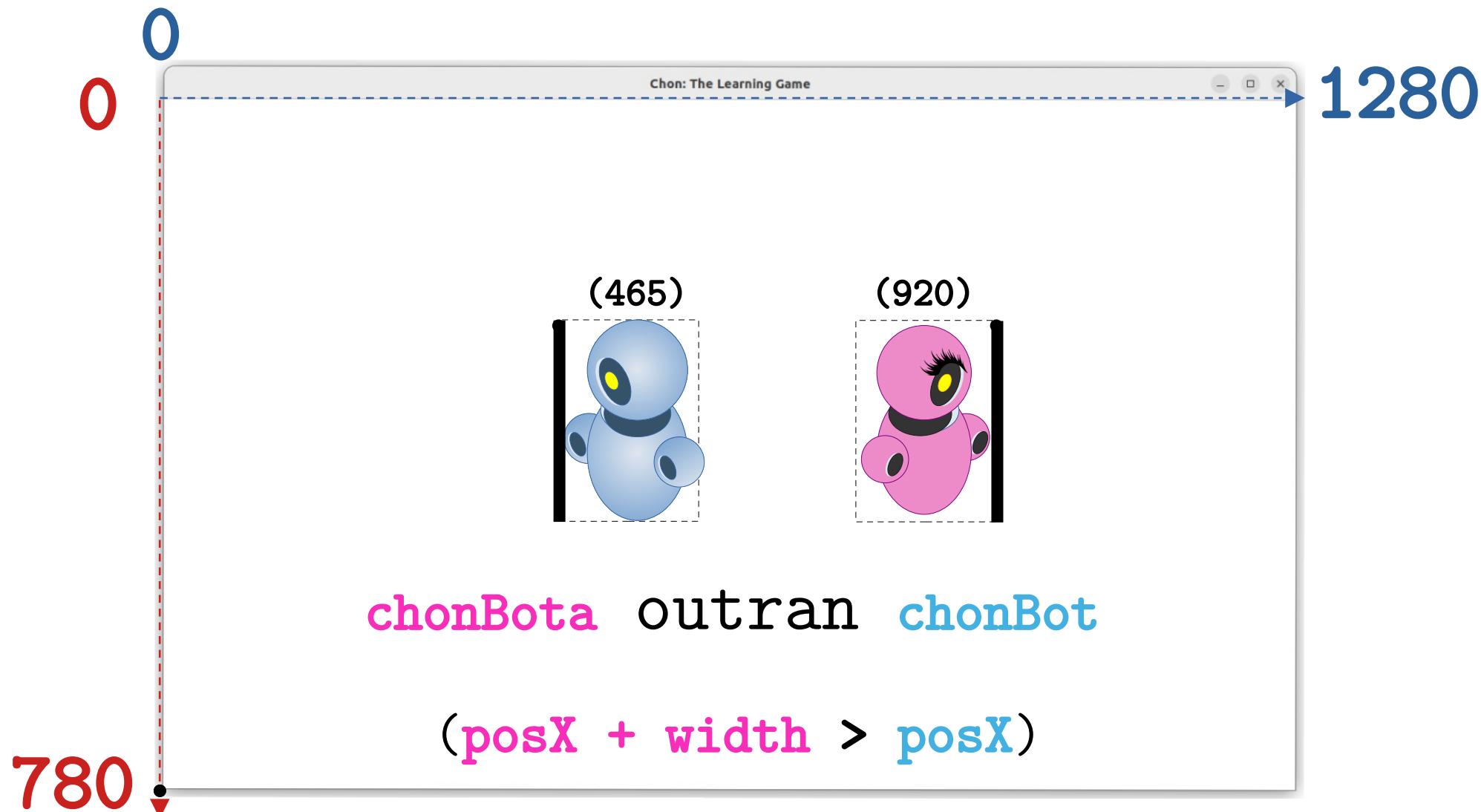
# Axis X: Condition 2



# Axis X: Condition 2



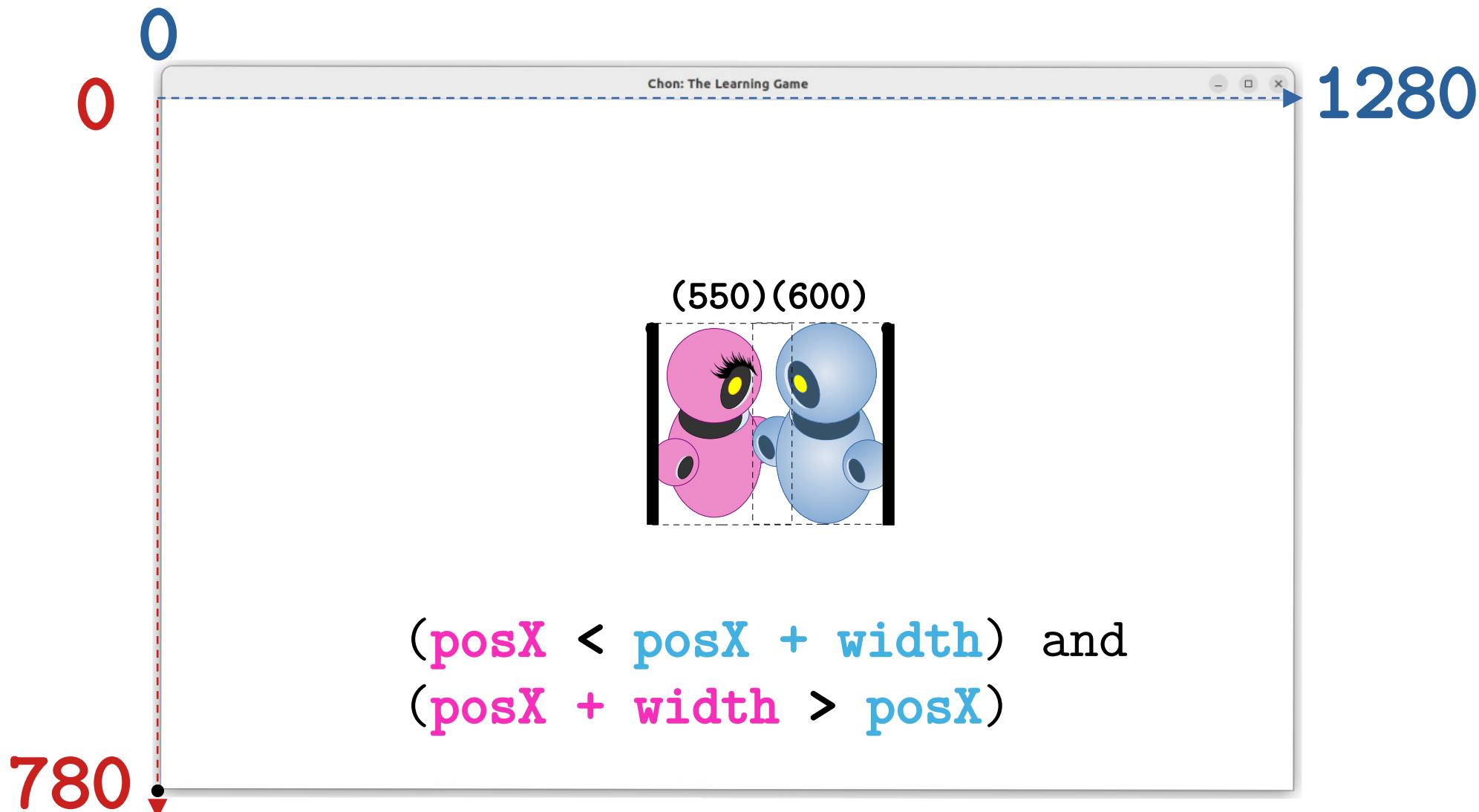
# Axis X: Condition 2



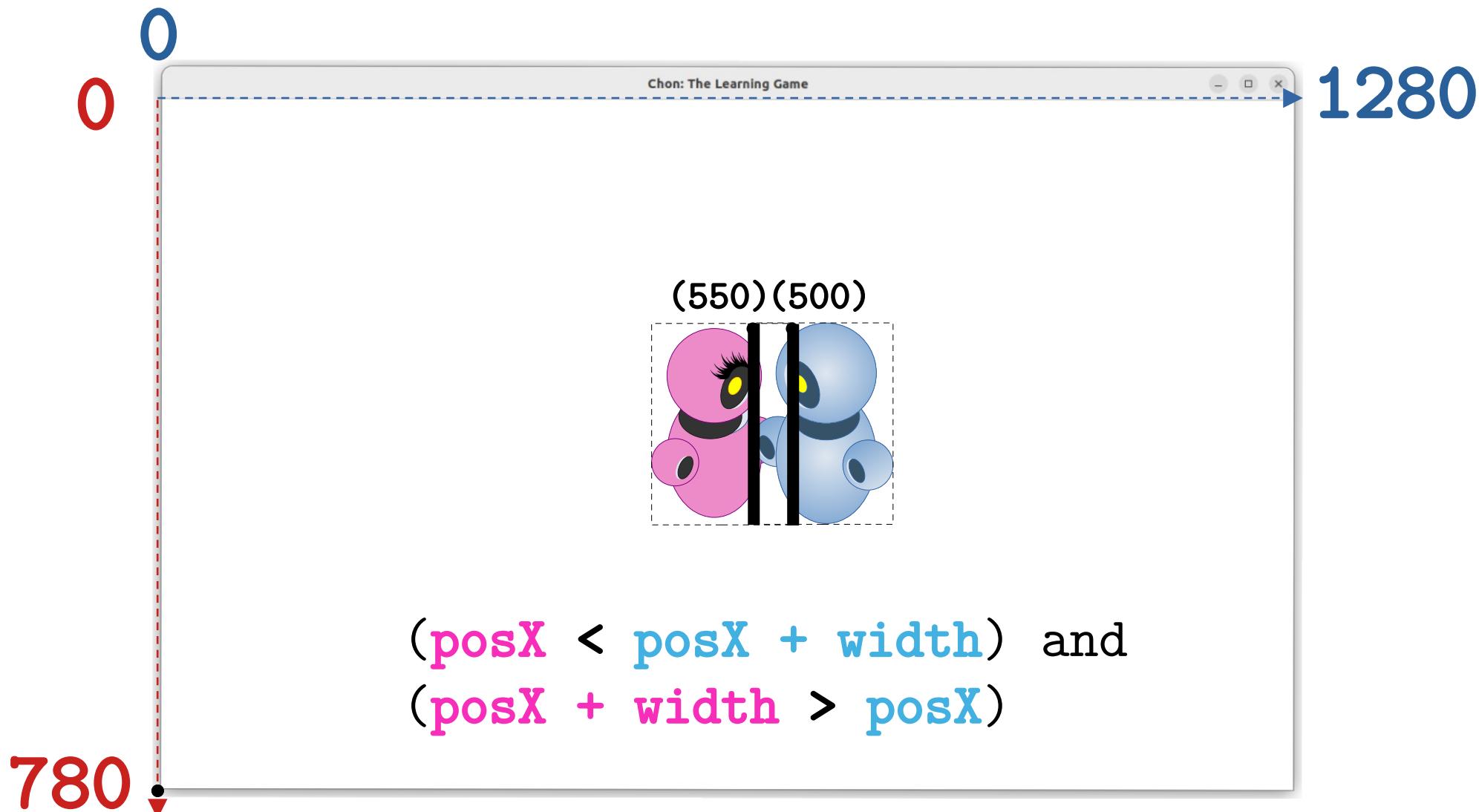
## Axis X: Condition 1 And 2

The **protagonist** must **cross** other agents  
and do not **outrun** them.

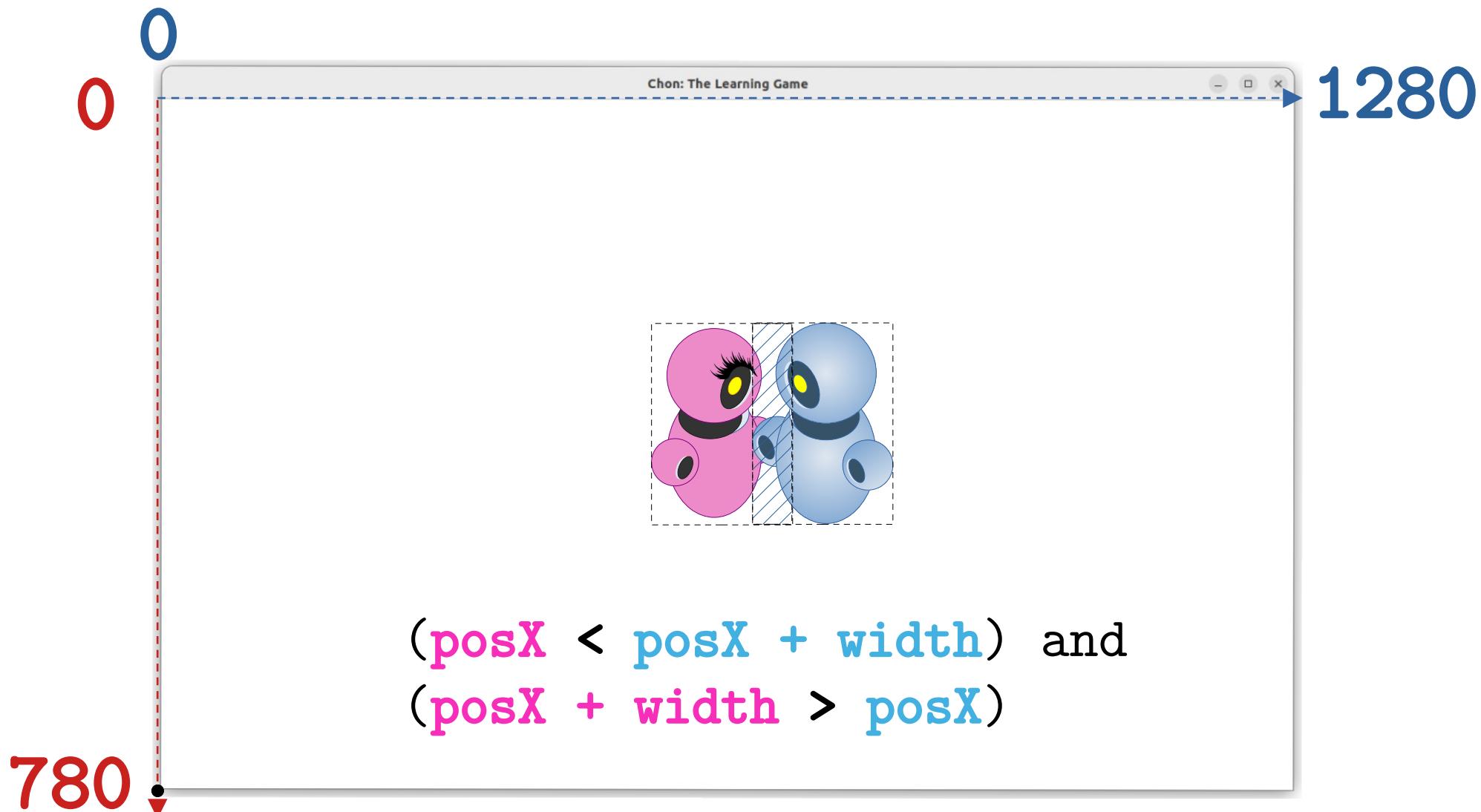
# Axis X: Condition 1 And 2



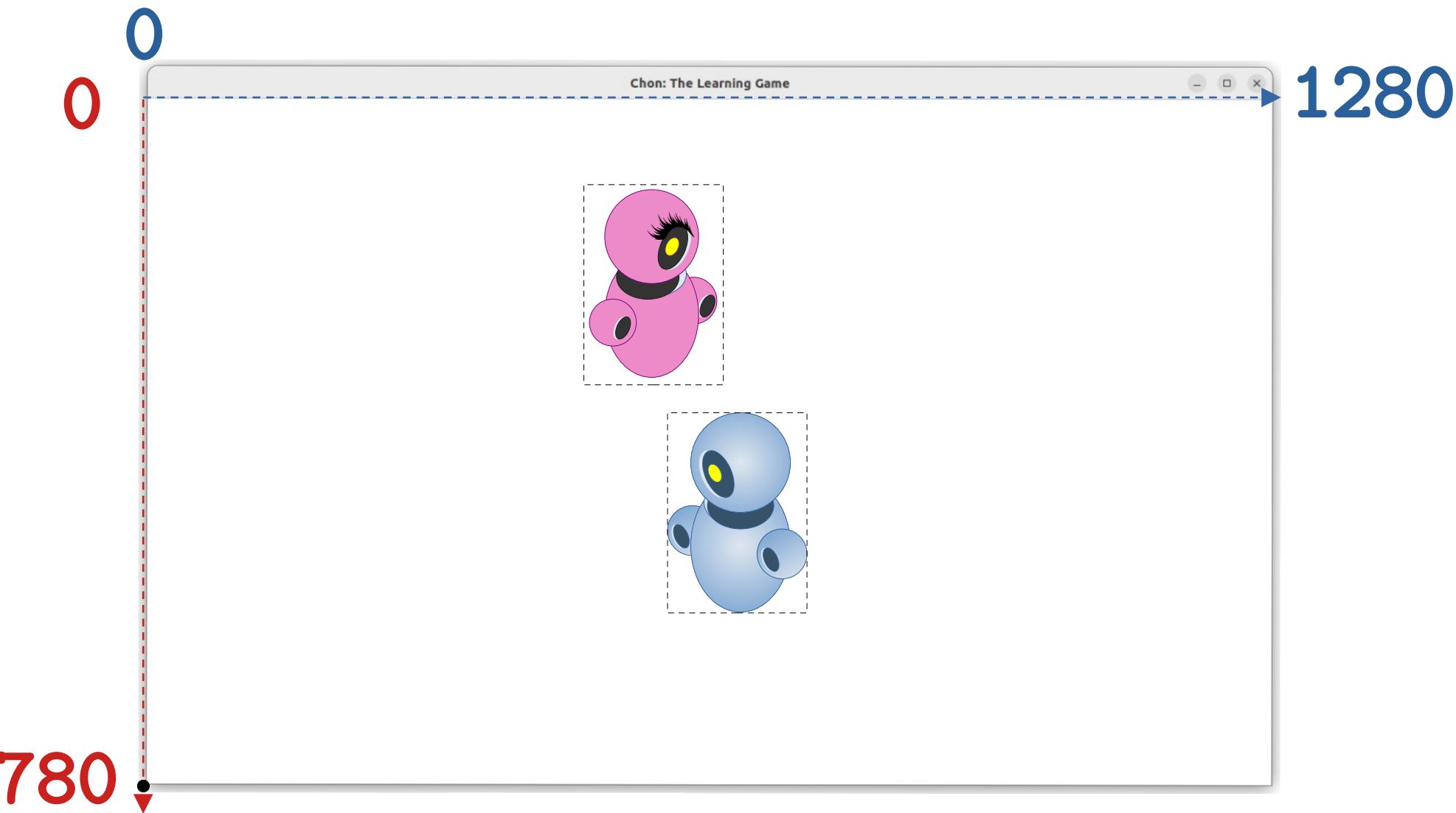
# Axis X: Condition 1 And 2



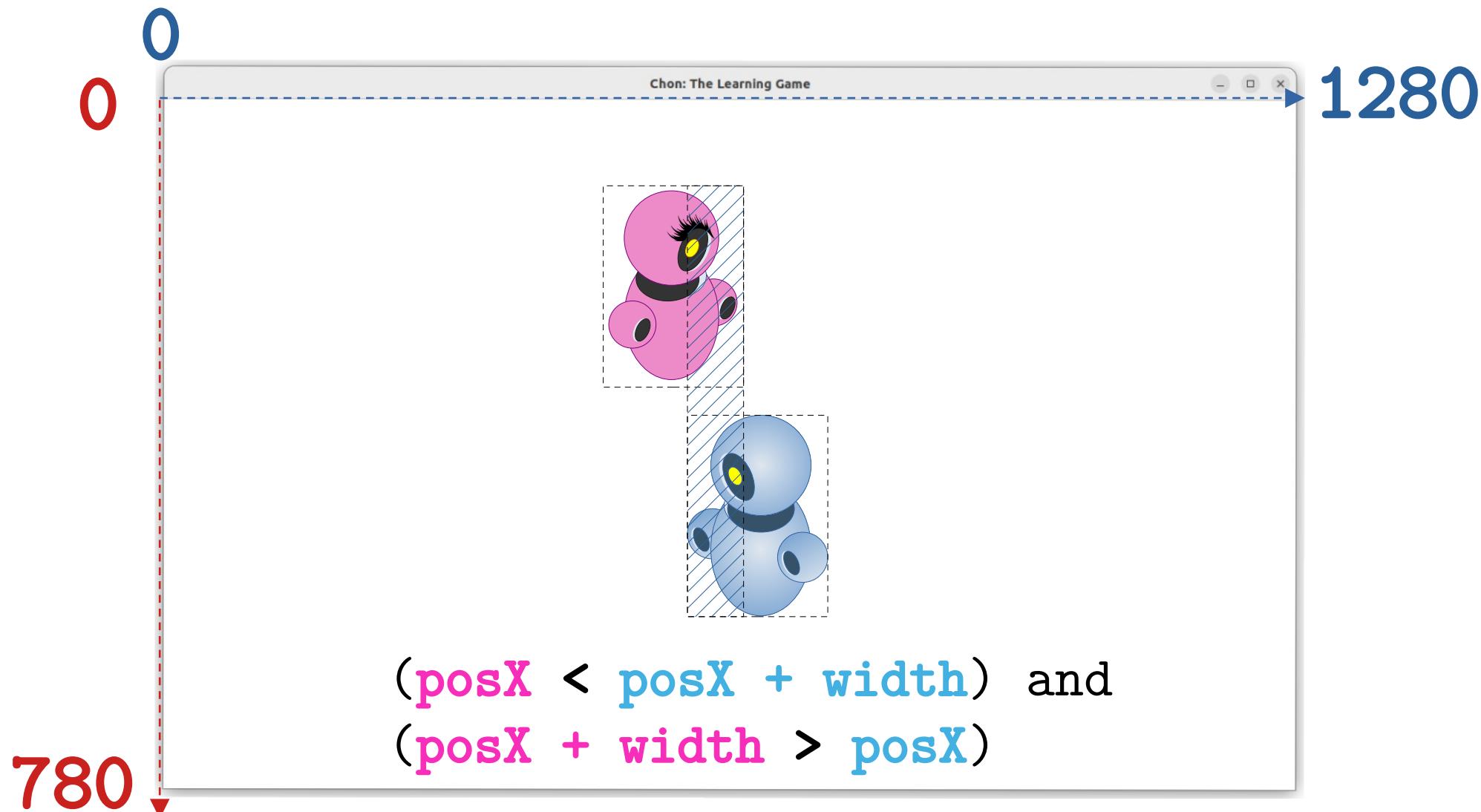
# Axis X: Condition 1 And 2



# Axis Y



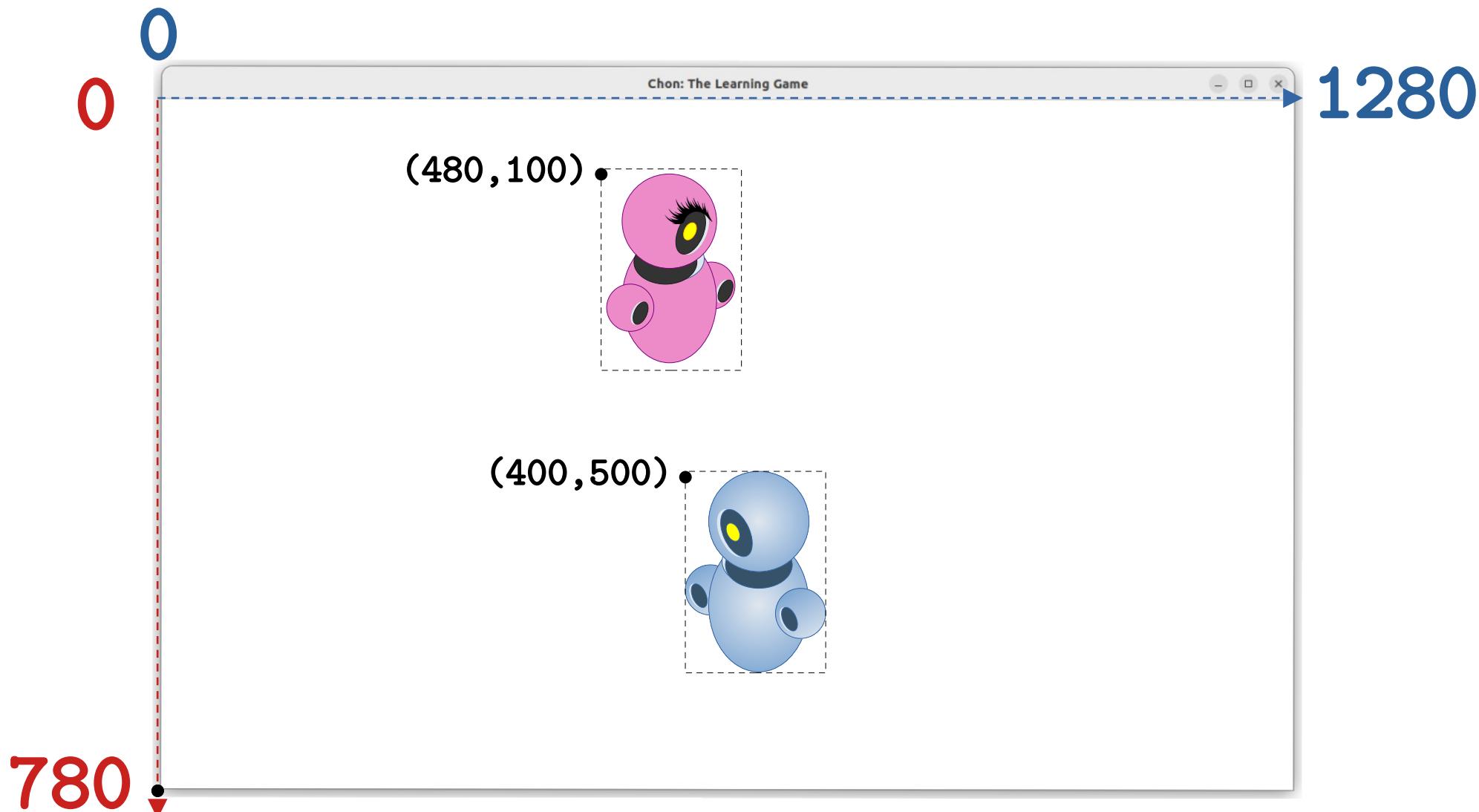
# Axis Y



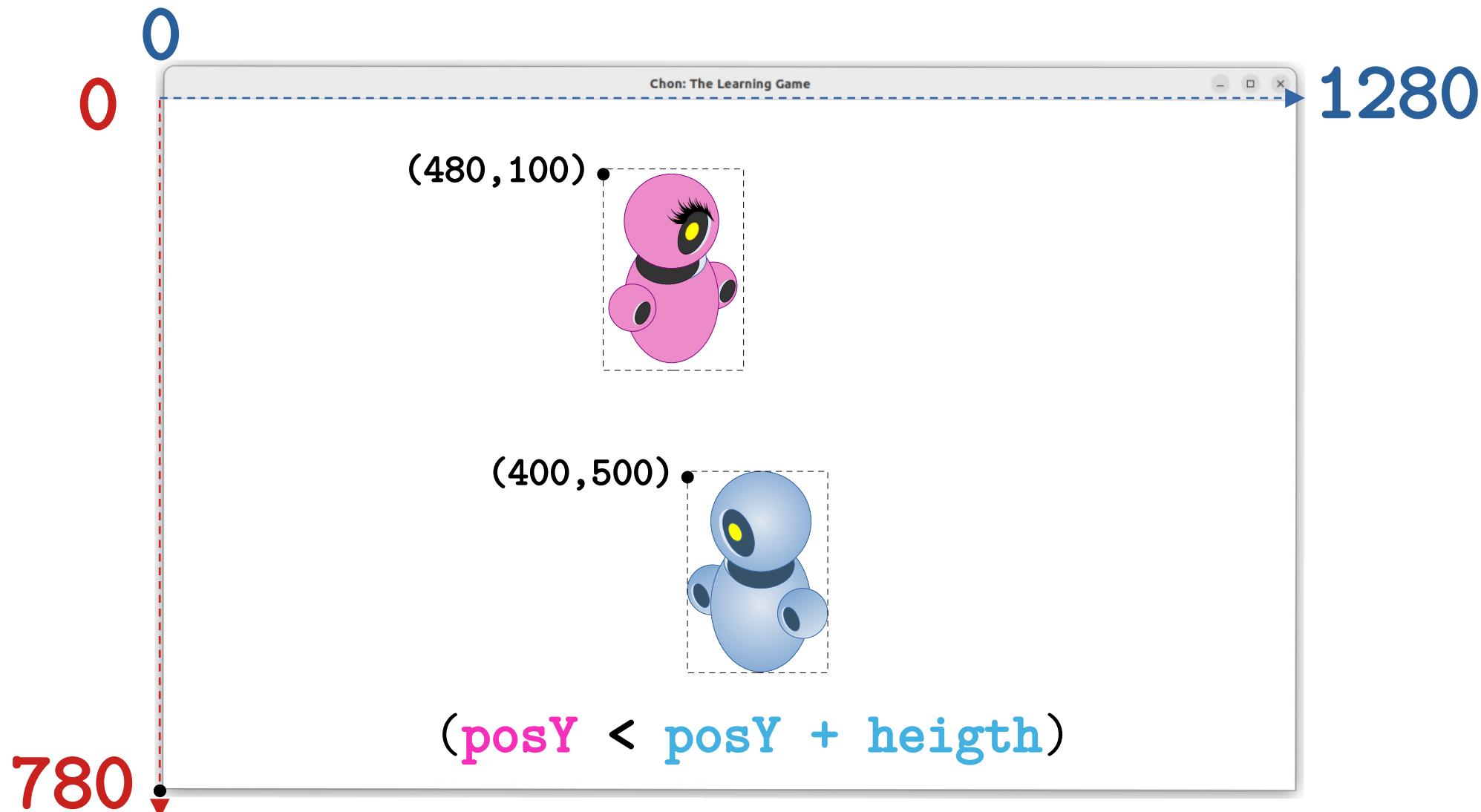
## Axis Y: Condition 3

The **protagonist** must not **outrun** the other agents.

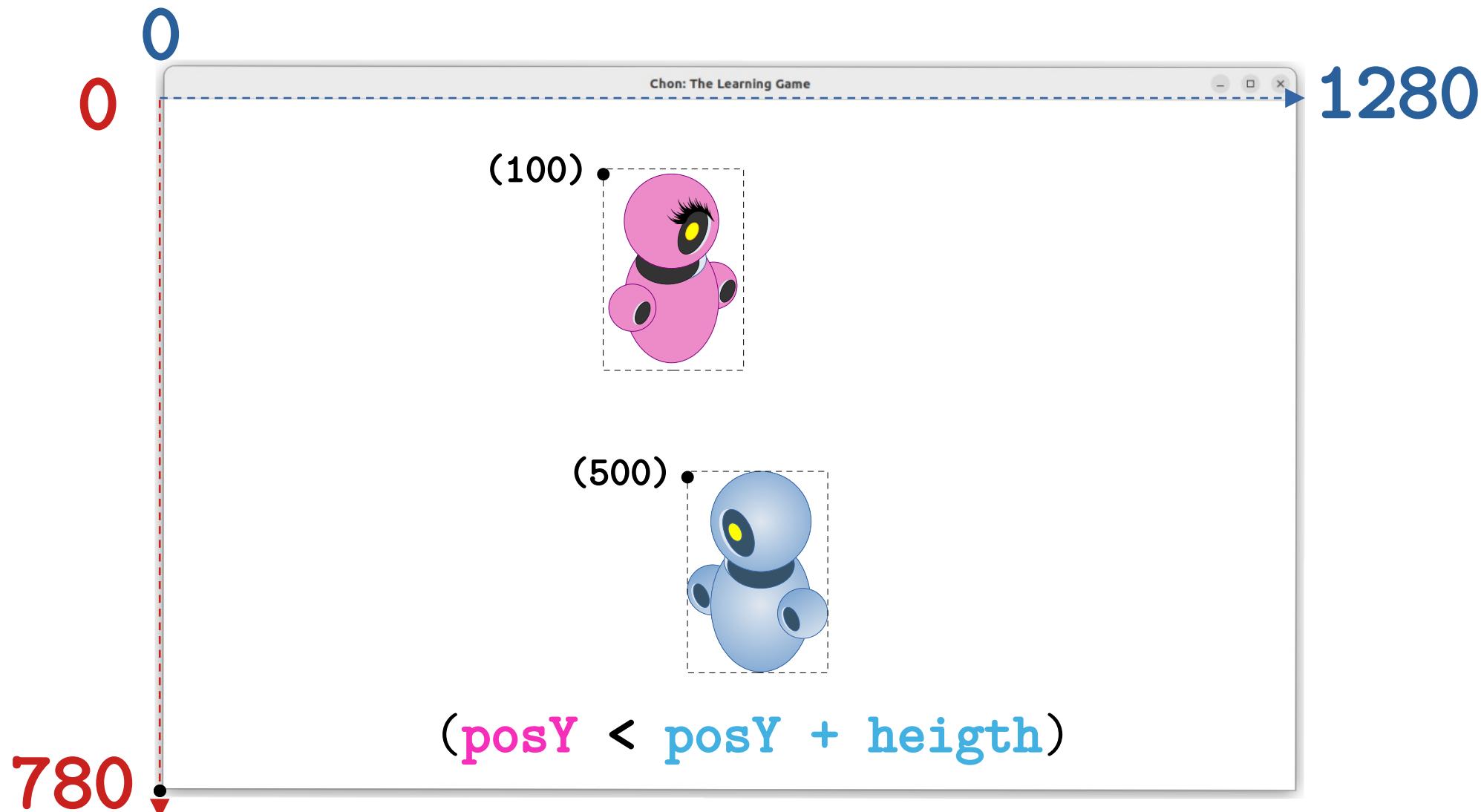
# Axis Y: Condition 3



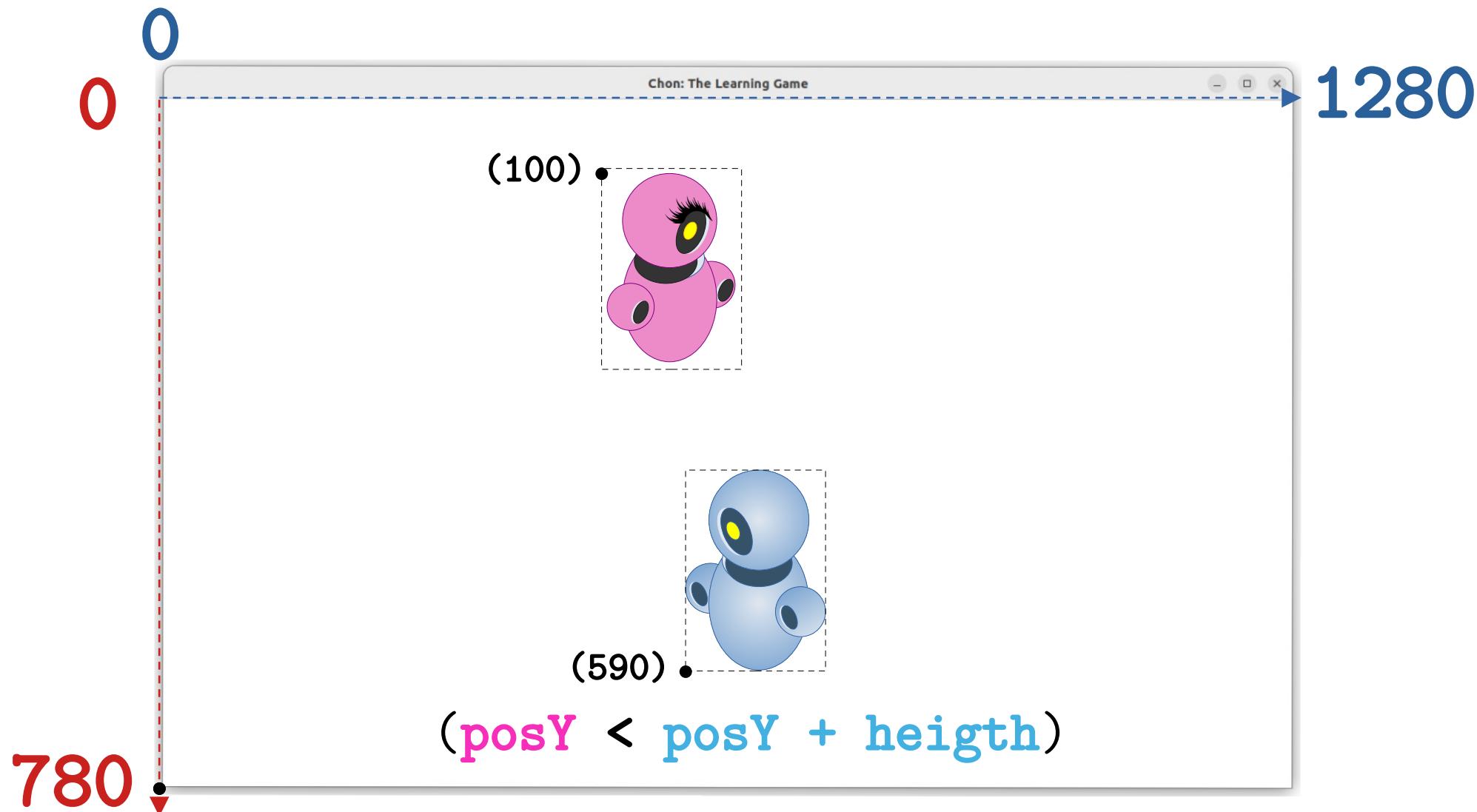
# Axis Y: Condition 3



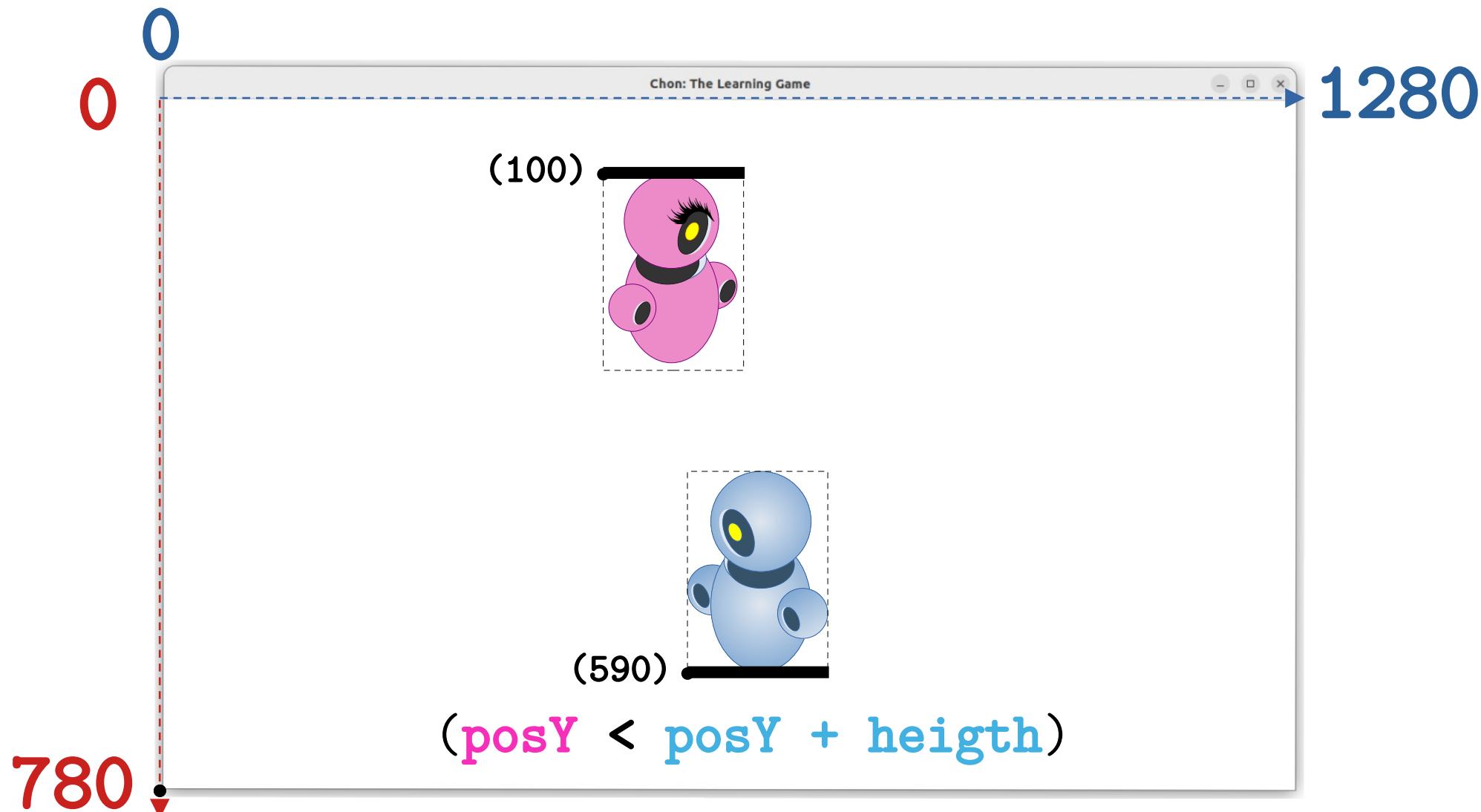
# Axis Y: Condition 3



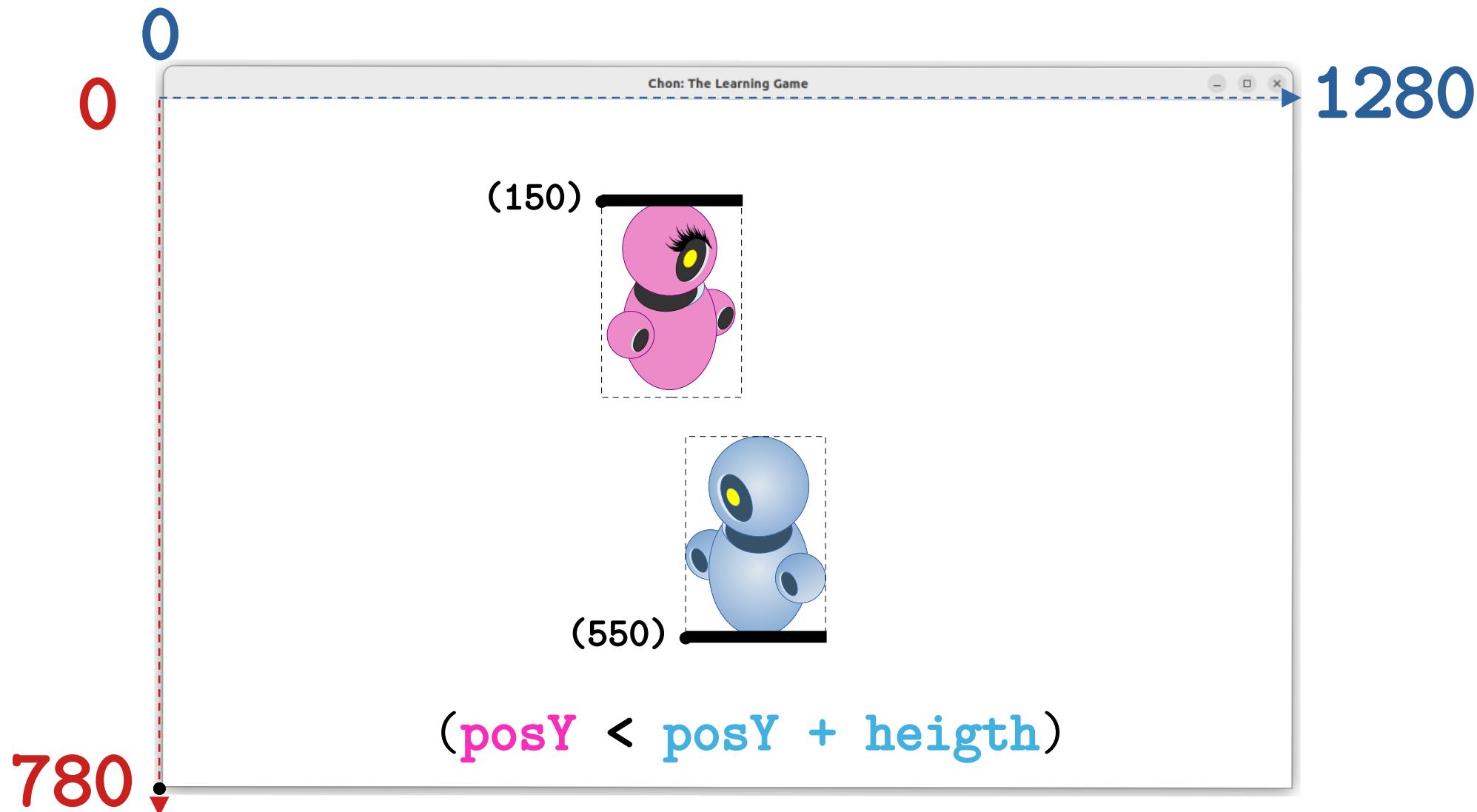
# Axis Y: Condition 3



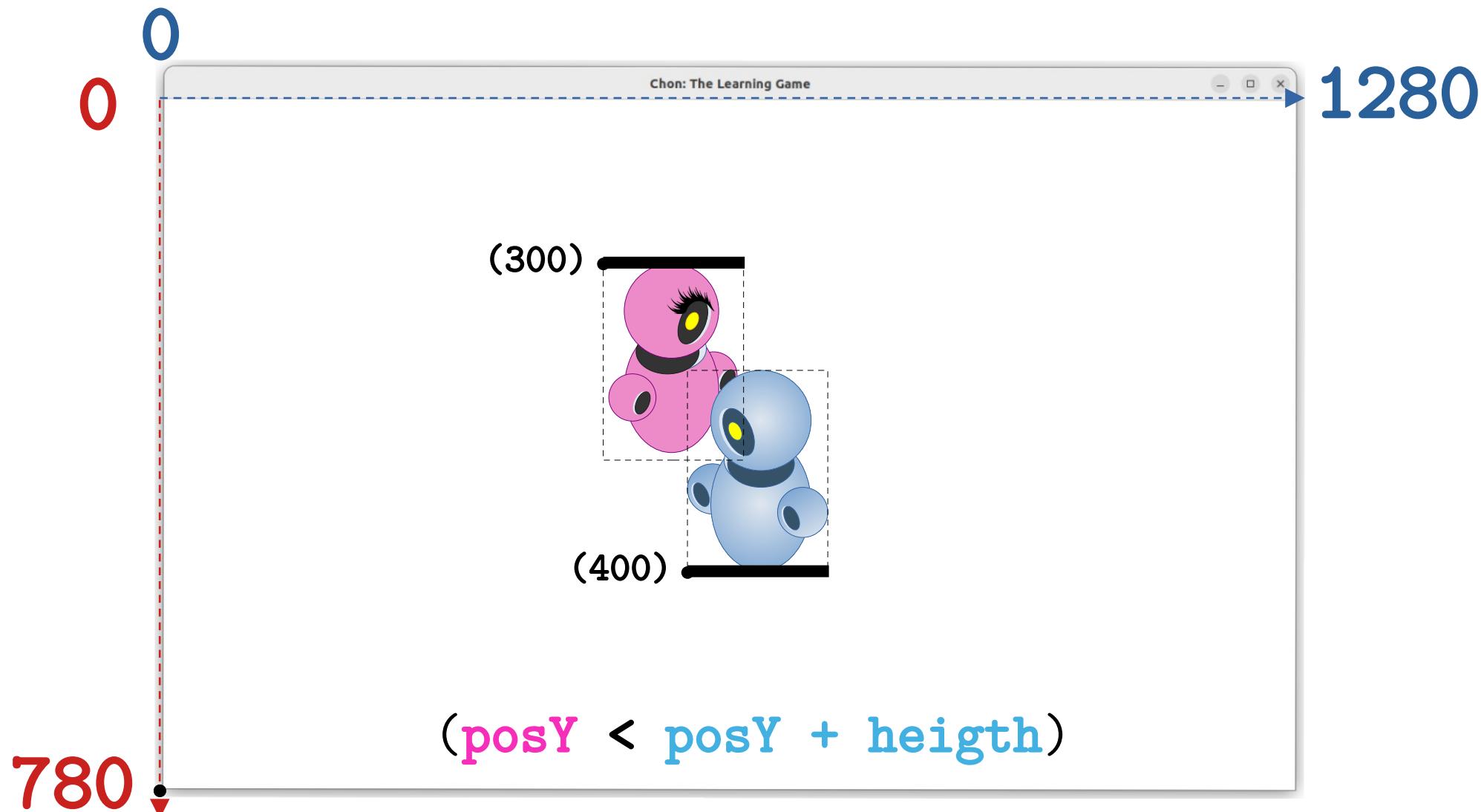
# Axis Y: Condition 3



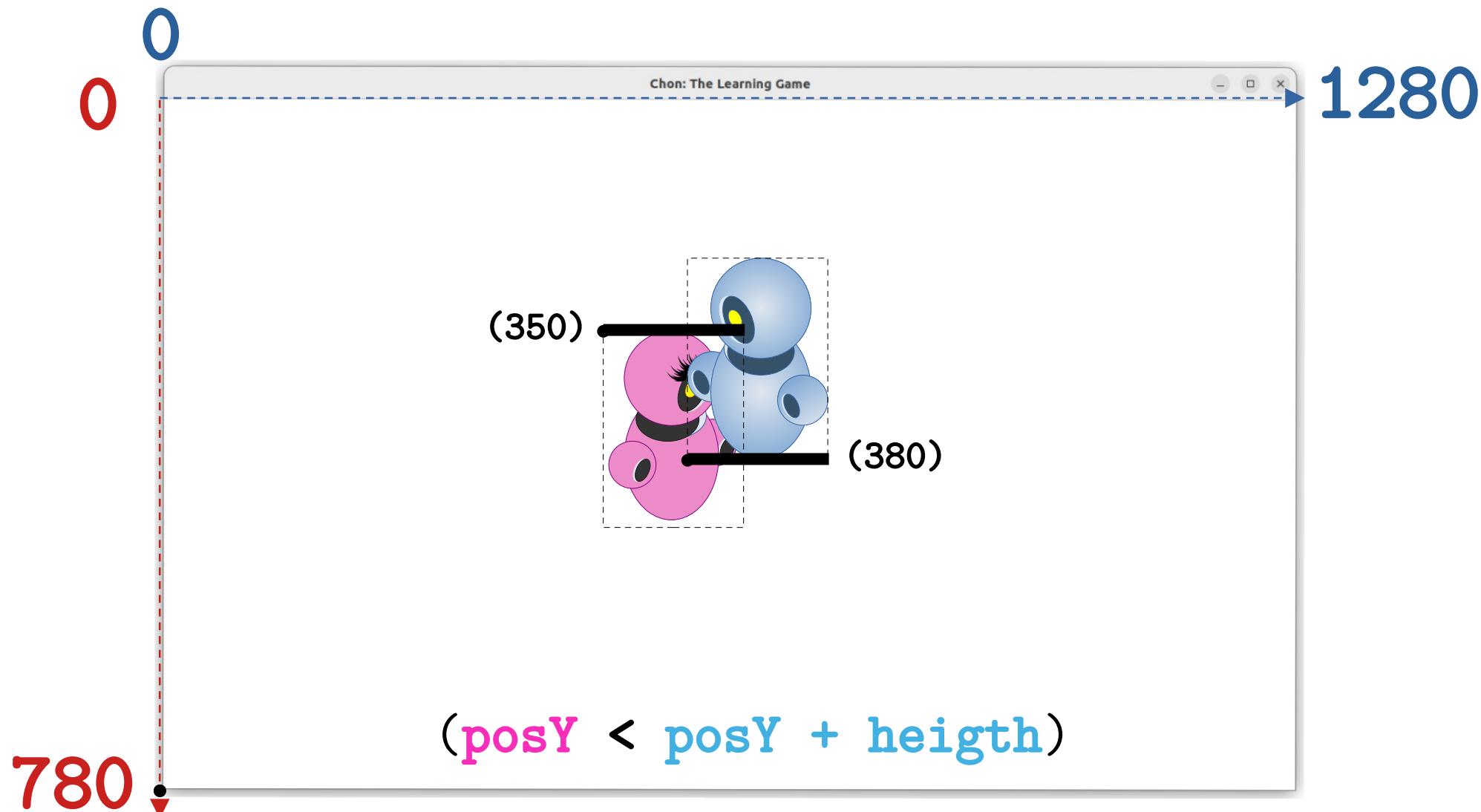
# Axis Y: Condition 3



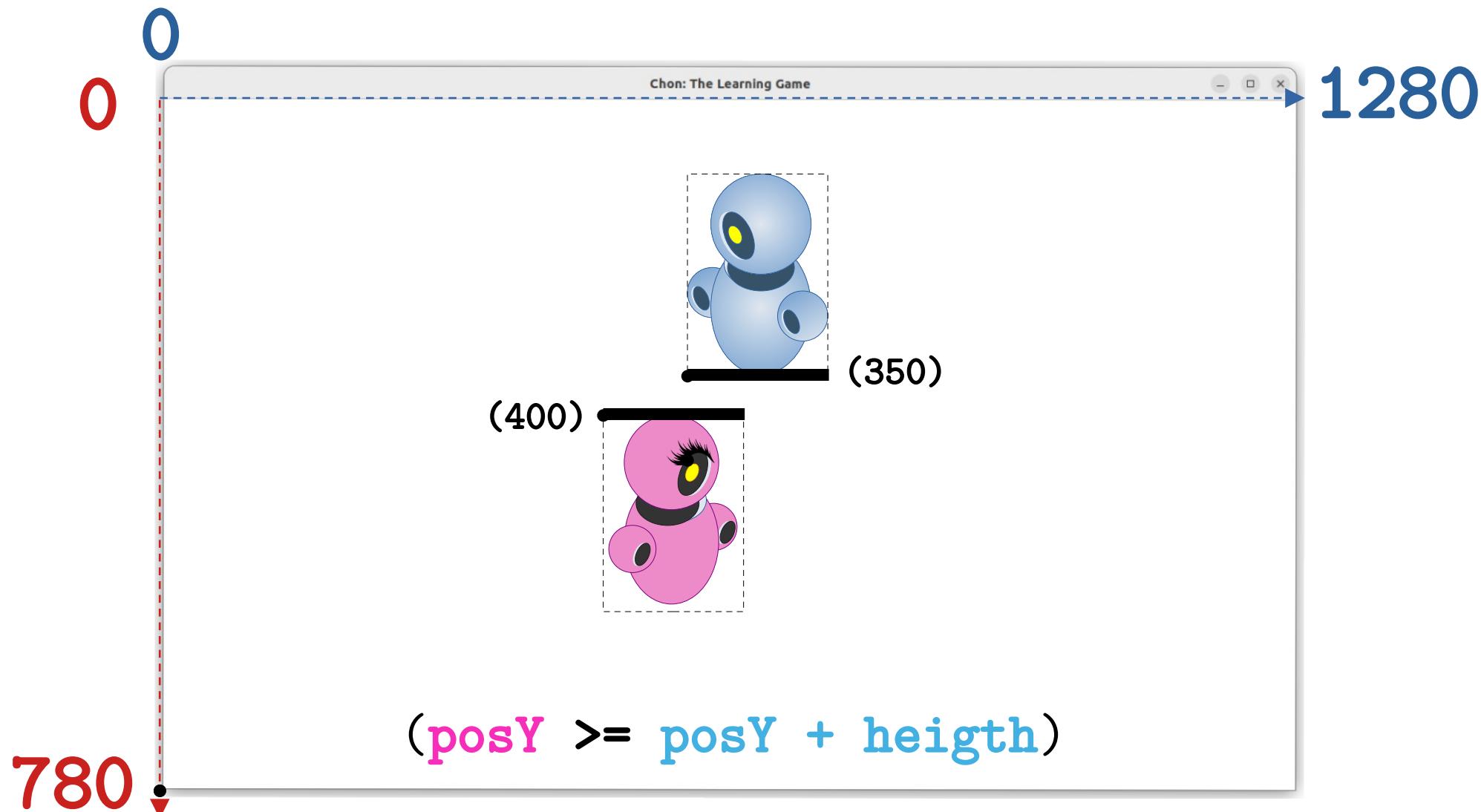
# Axis Y: Condition 3



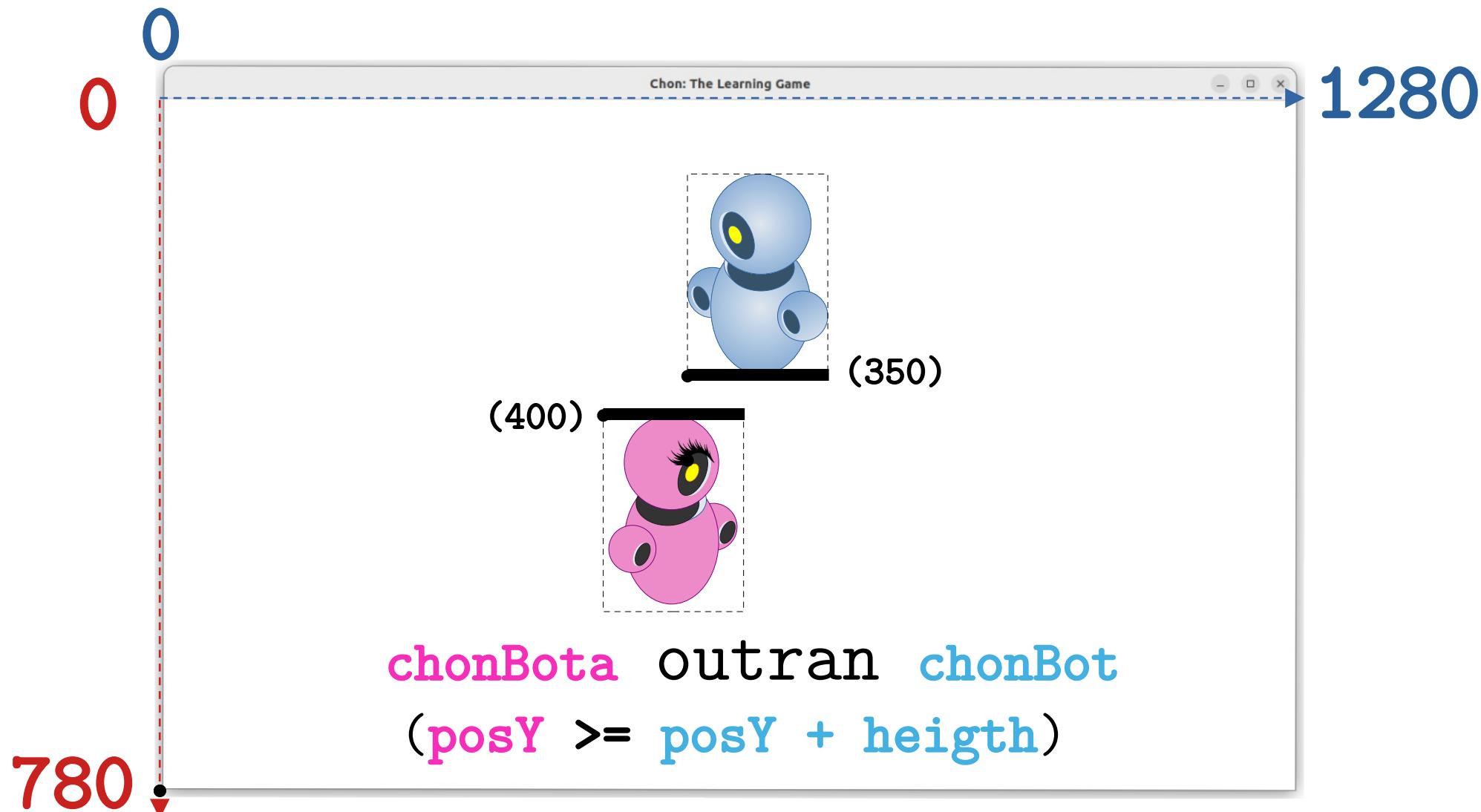
# Axis Y: Condition 3



# Axis Y: Condition 3



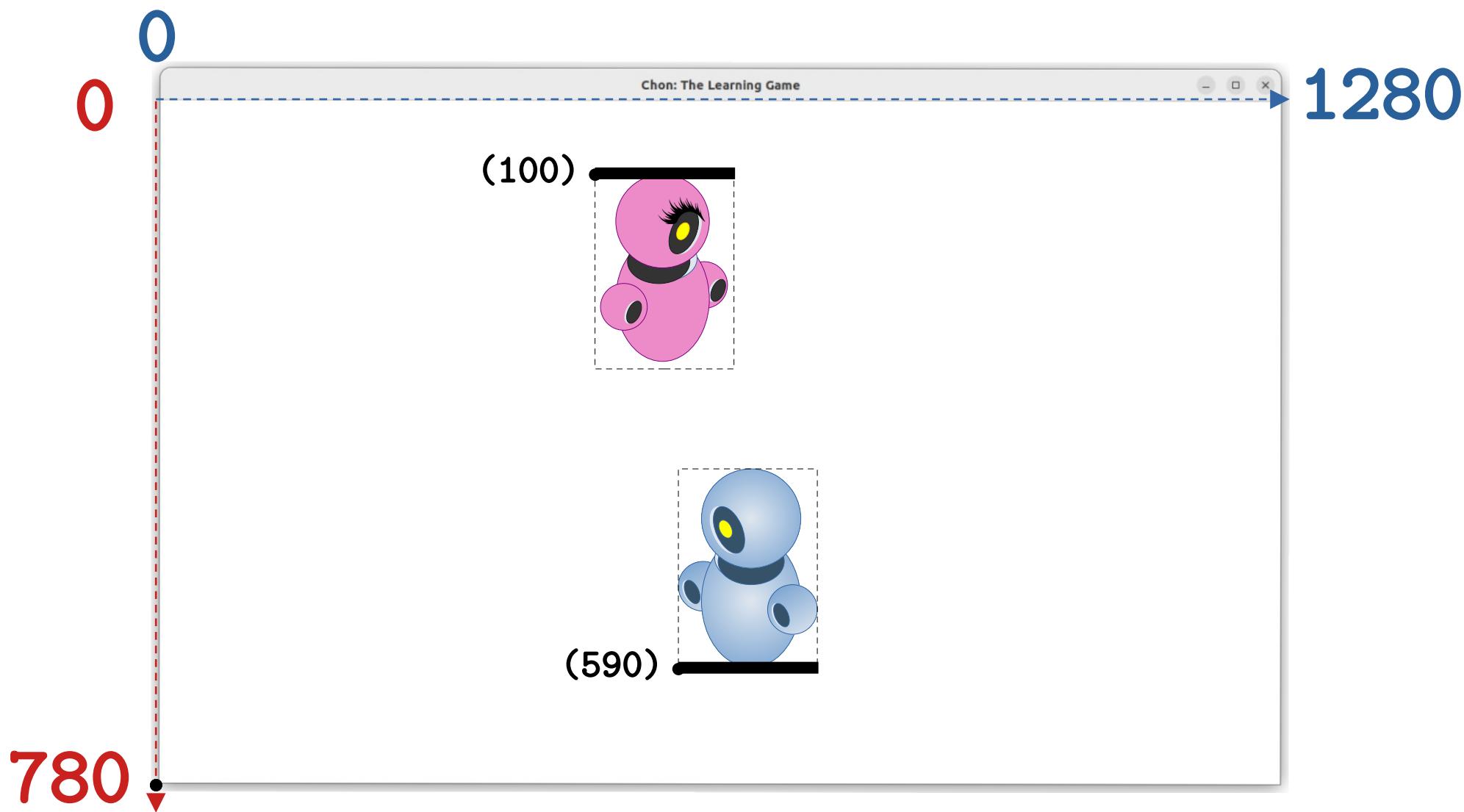
# Axis Y: Condition 3



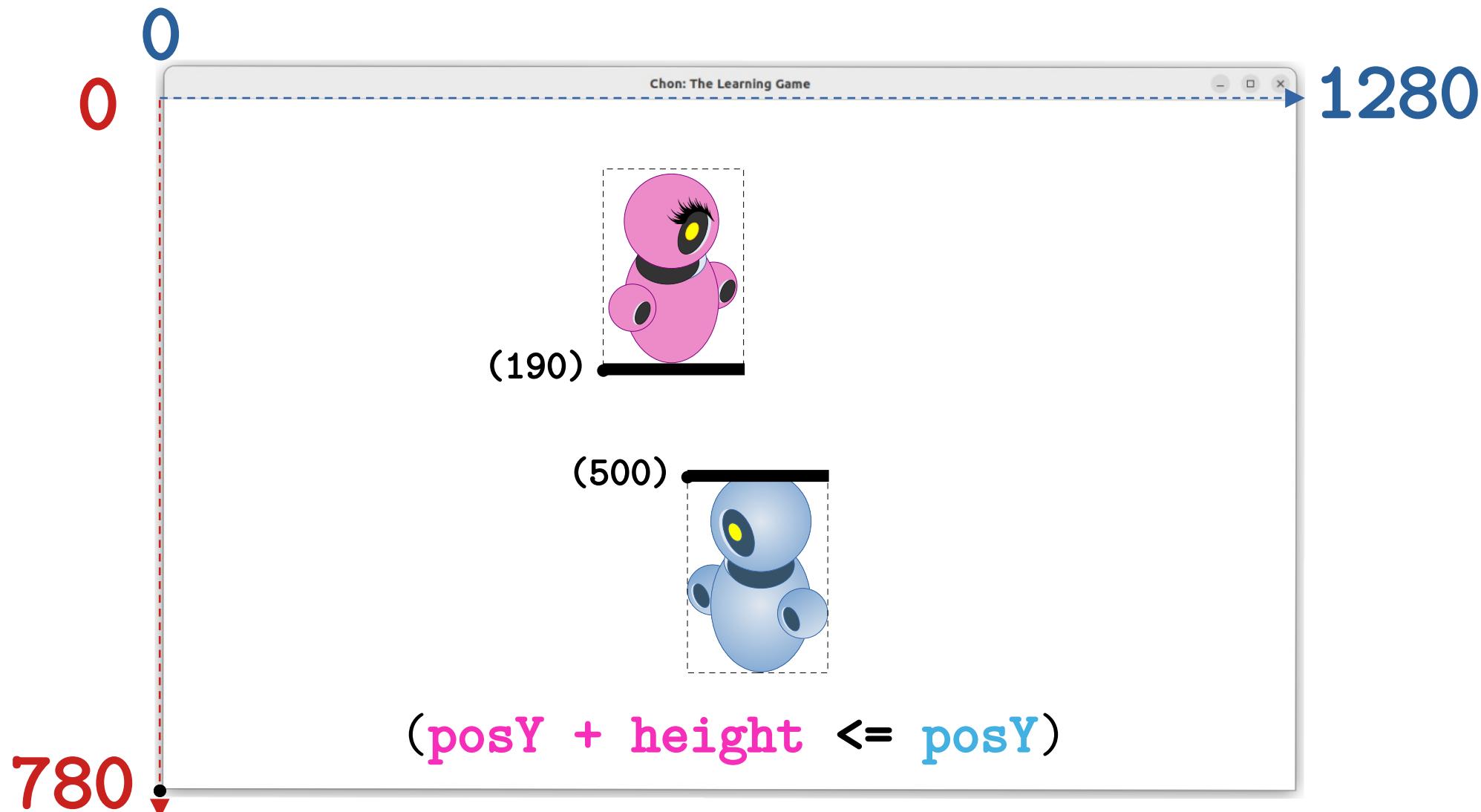
## Axis Y: Condition 4

The **protagonist** must **cross** the other agents.

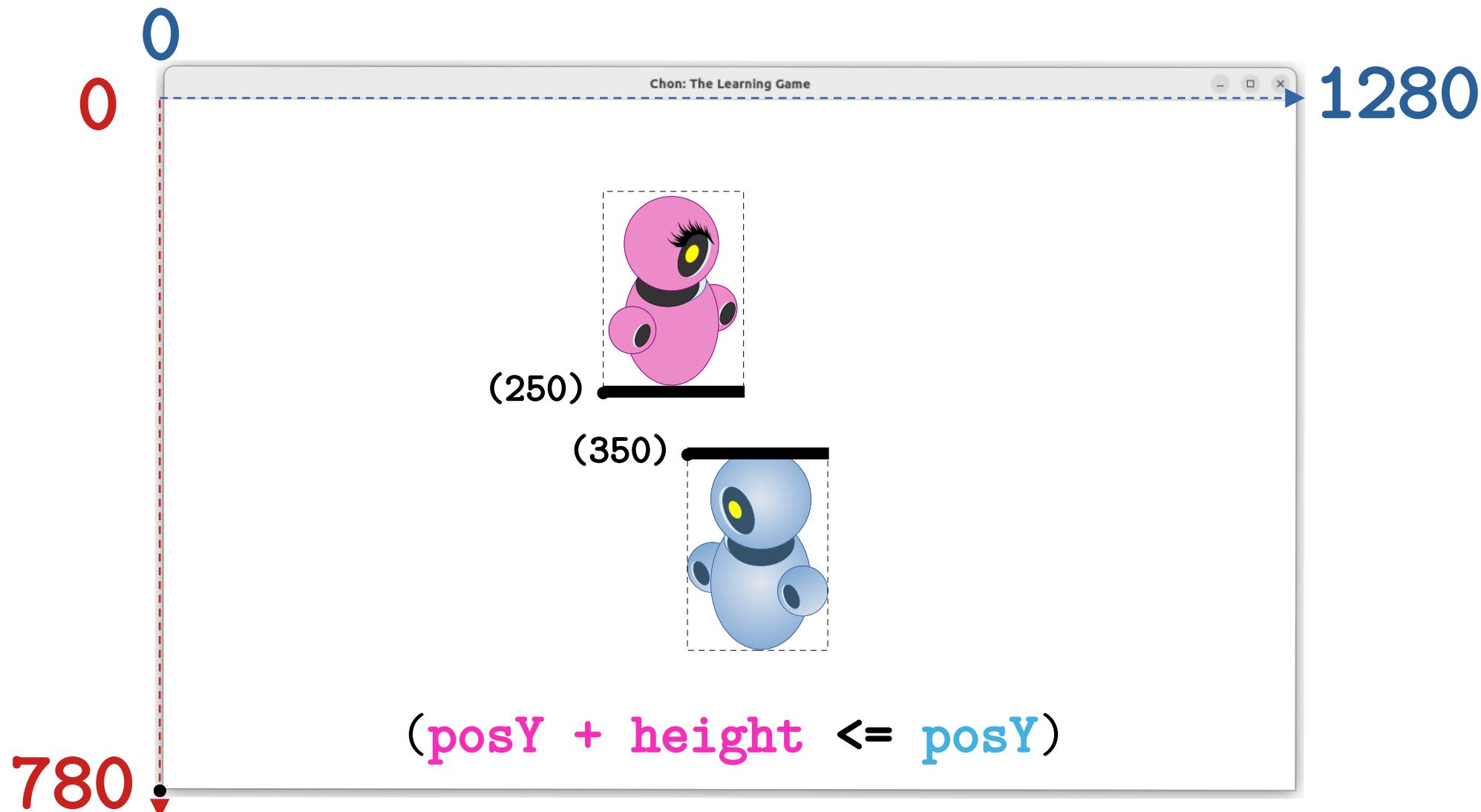
# Axis Y: Condition 4



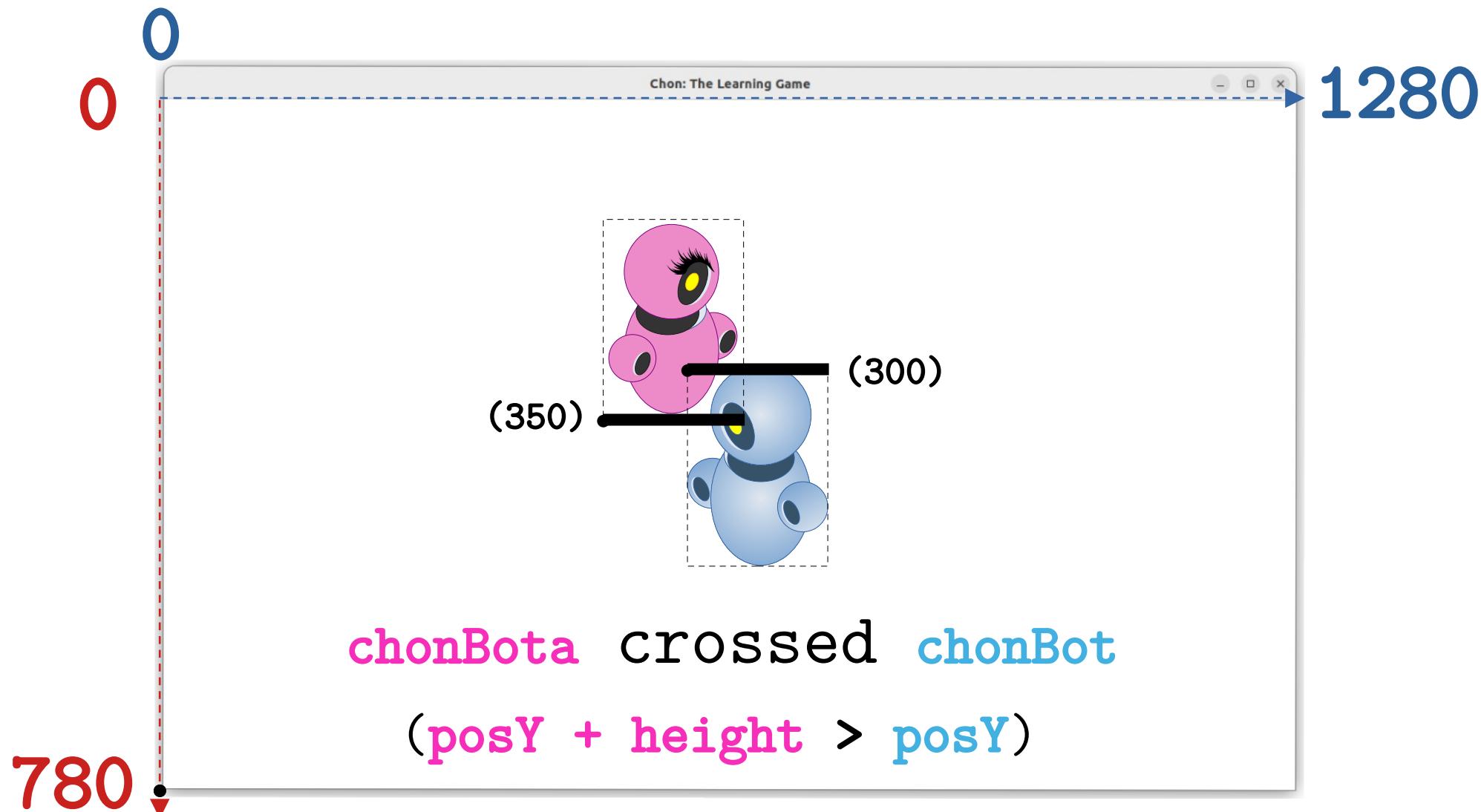
# Axis Y: Condition 4



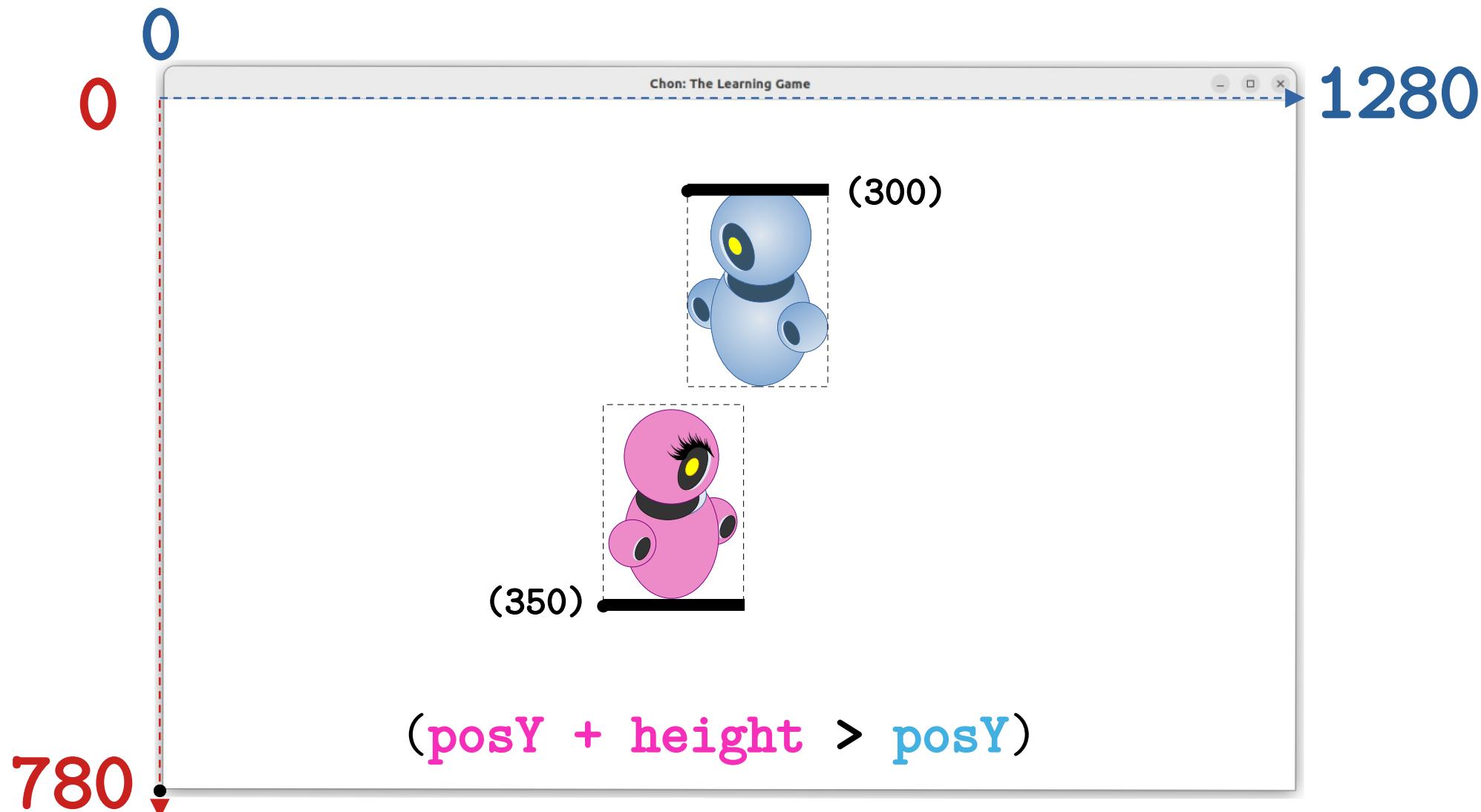
# Axis Y: Condition 4



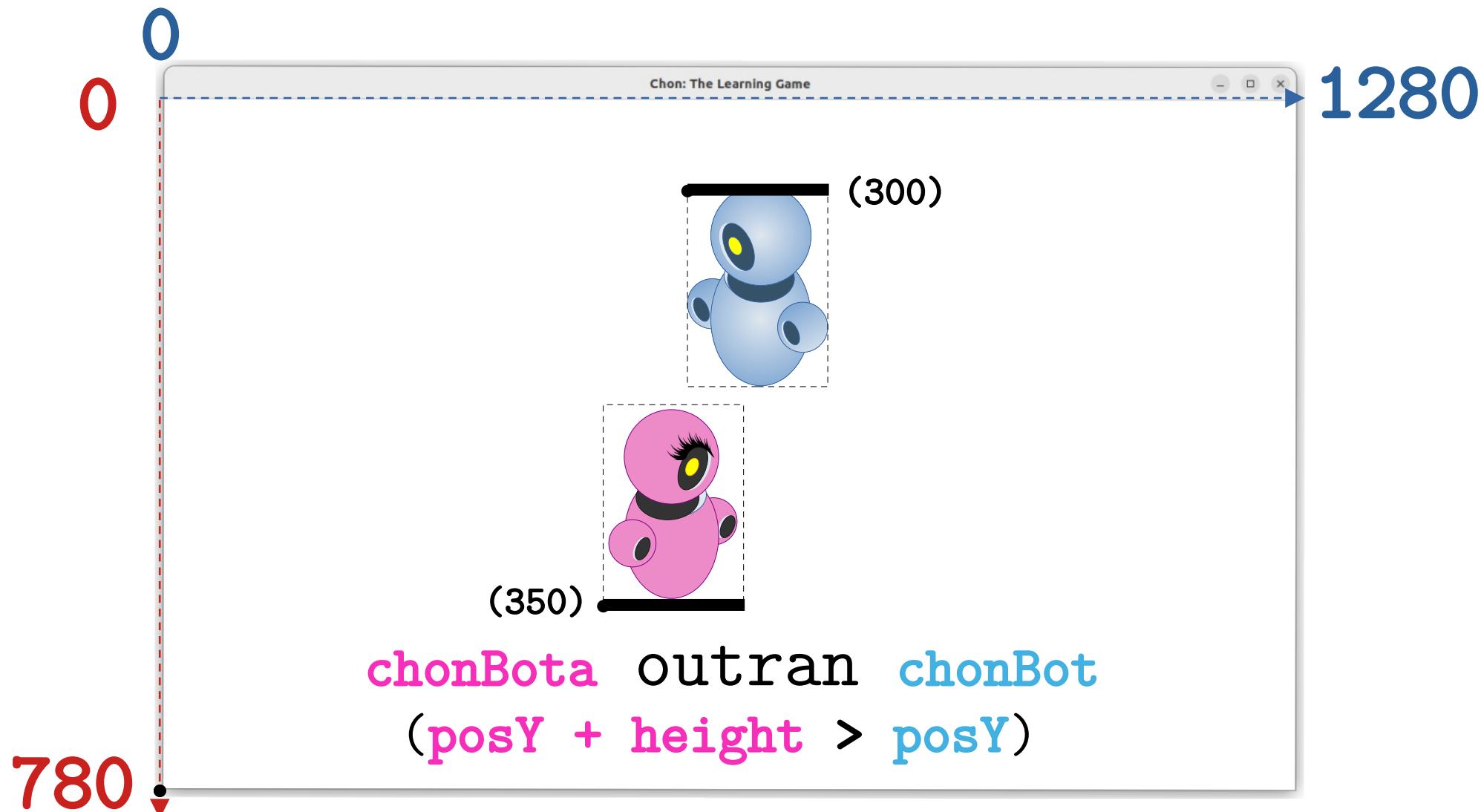
# Axis Y: Condition 4



# Axis Y: Condition 4



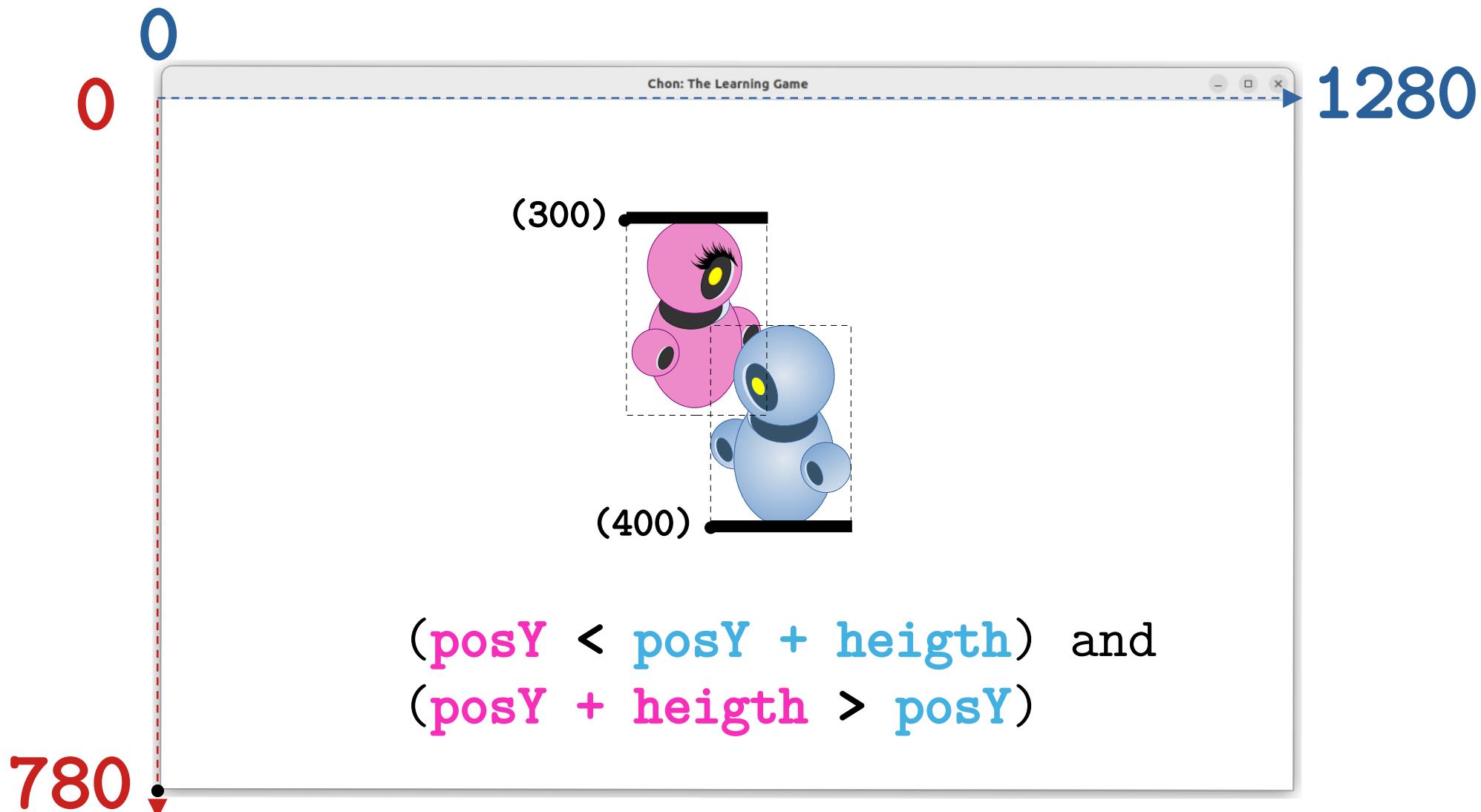
# Axis Y: Condition 4



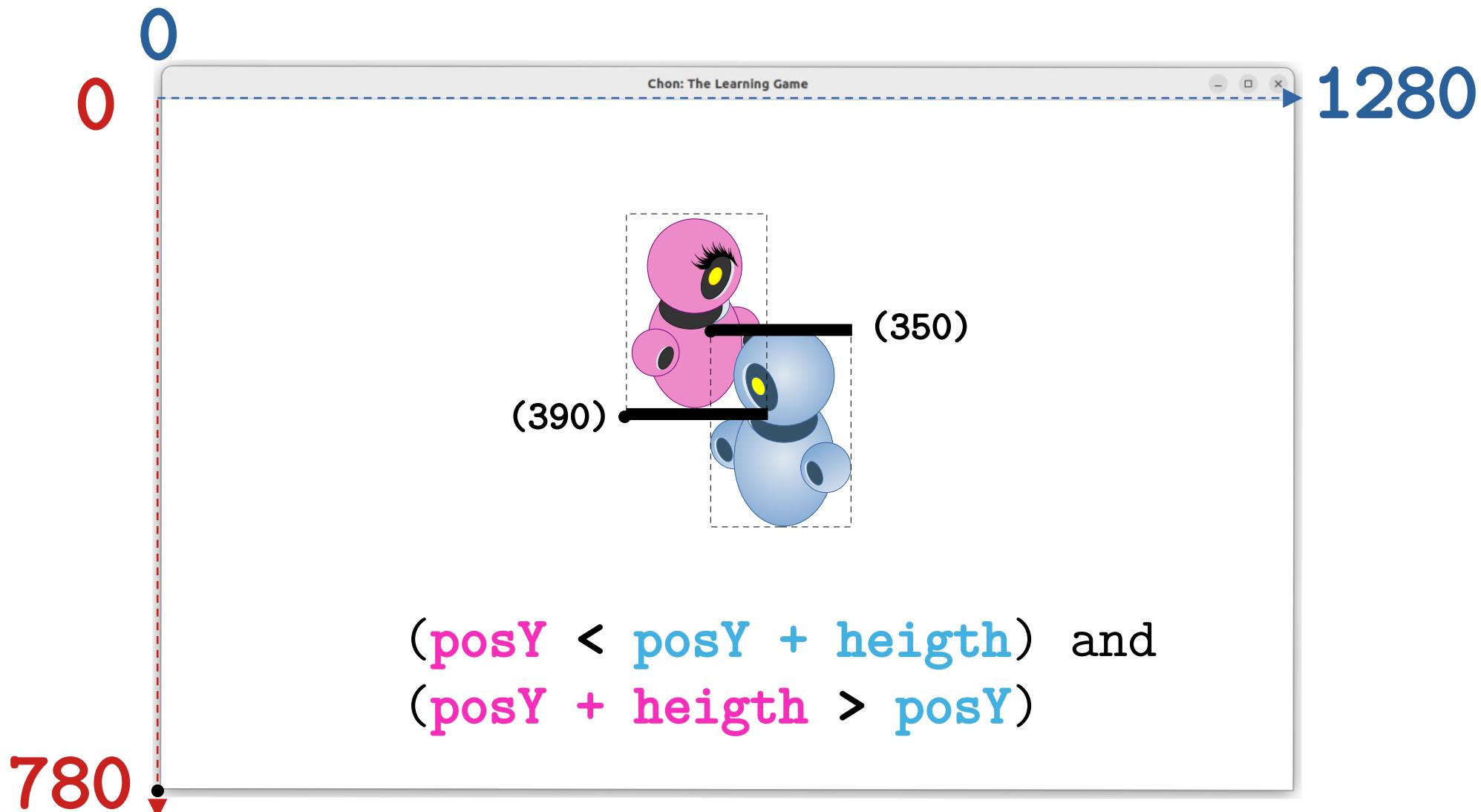
## Axis Y: Condition 3 And 4

The **protagonist** must **cross** other agents  
and do not **outrun** them.

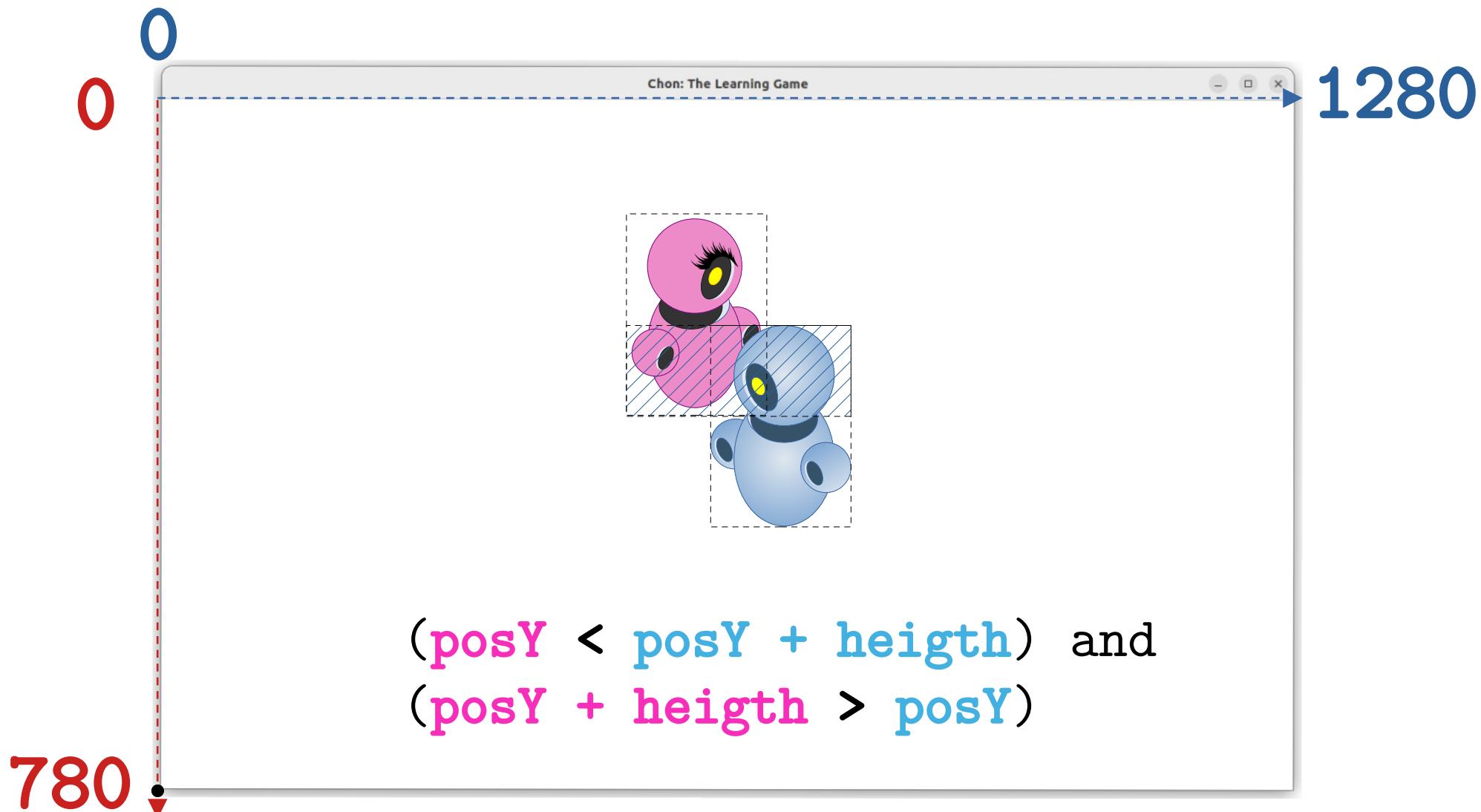
# Axis Y: Condition 3 And 4



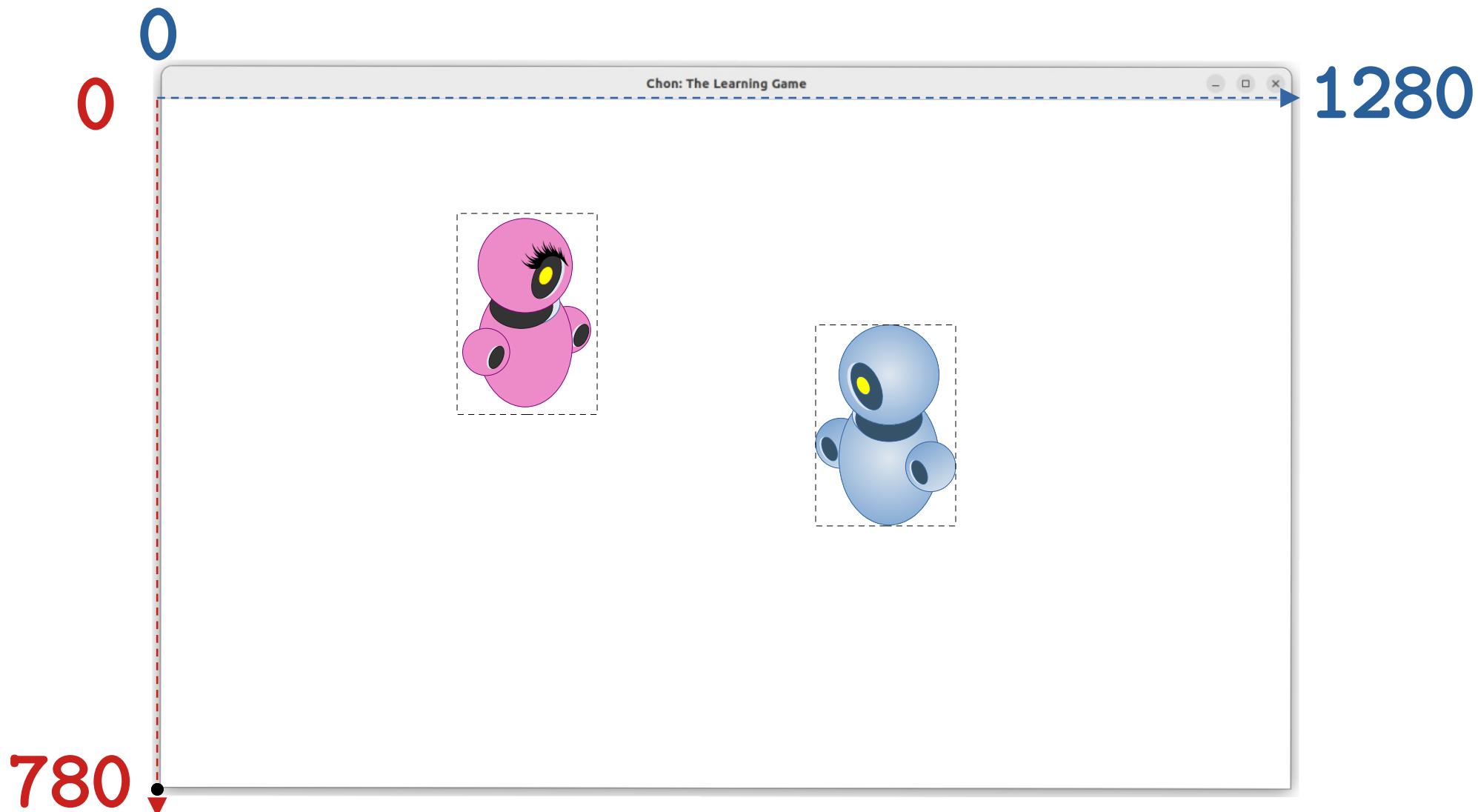
# Axis Y: Condition 3 And 4



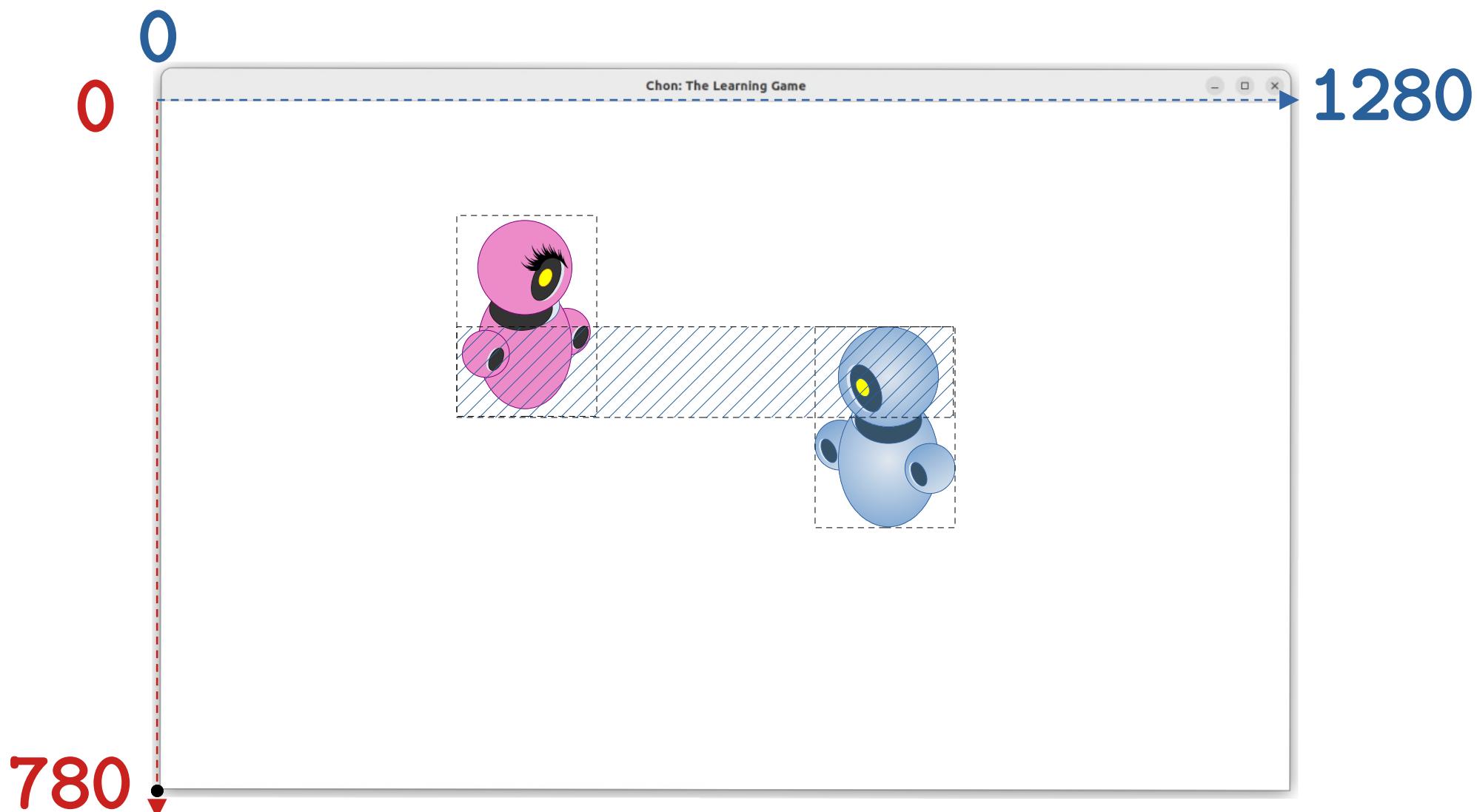
# Axis Y: Condition 3 And 4



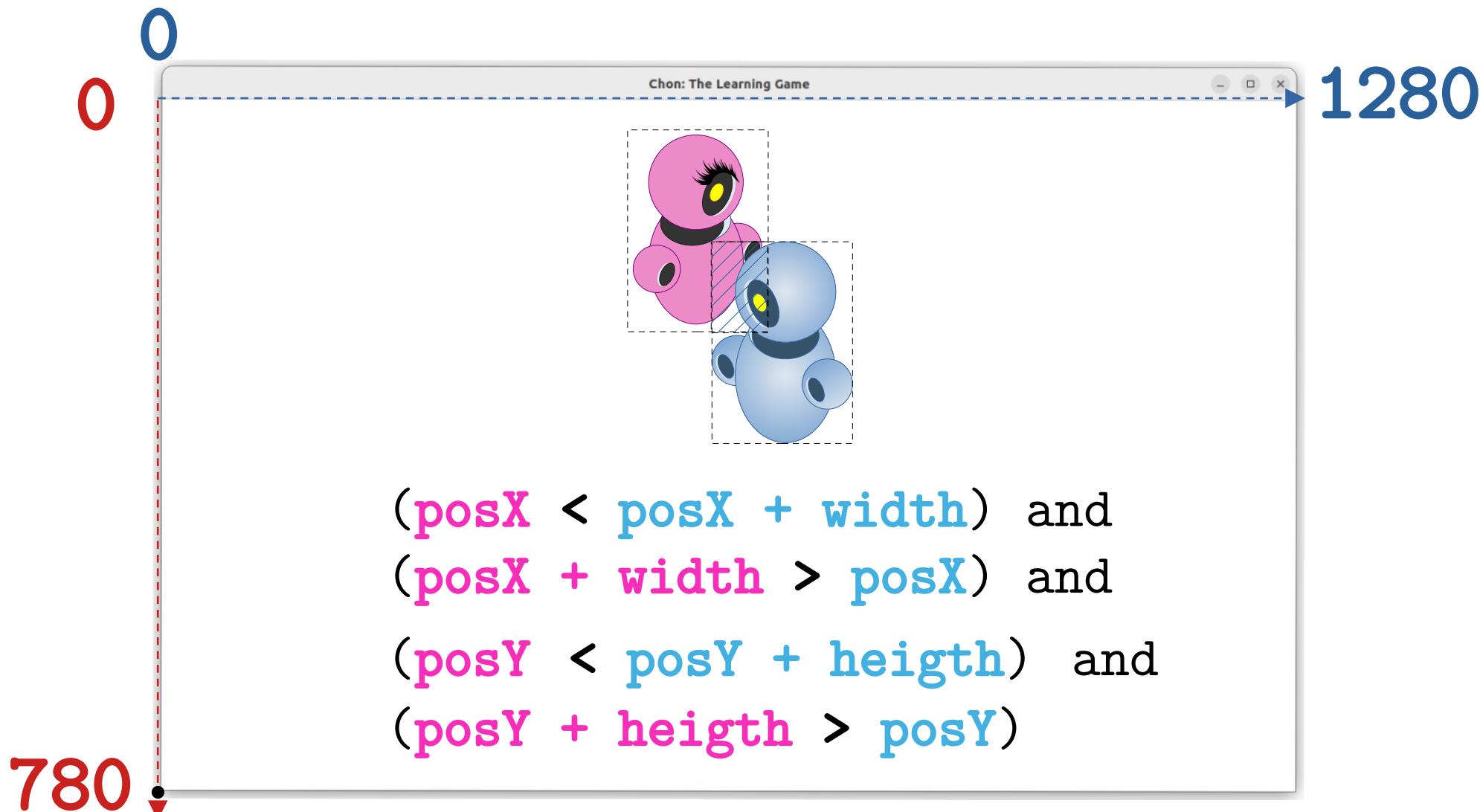
# Axis Y: Condition 3 And 4



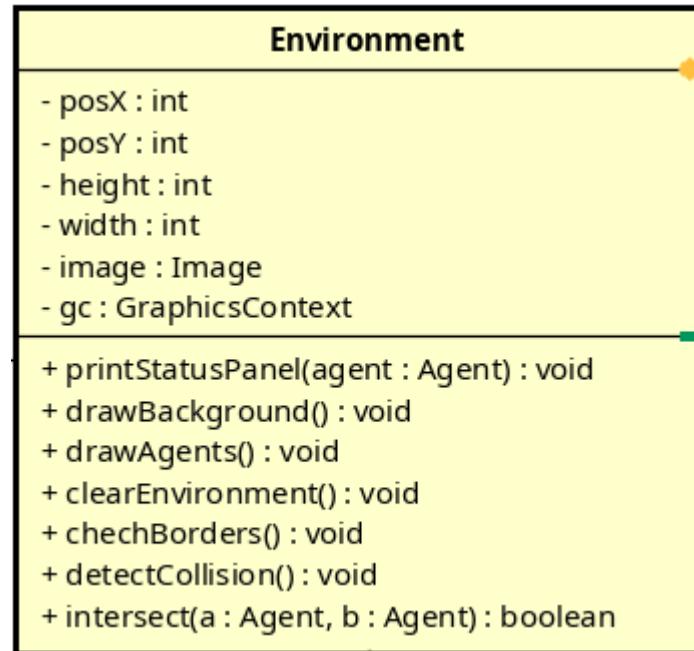
# Axis Y: Condition 3 And 4



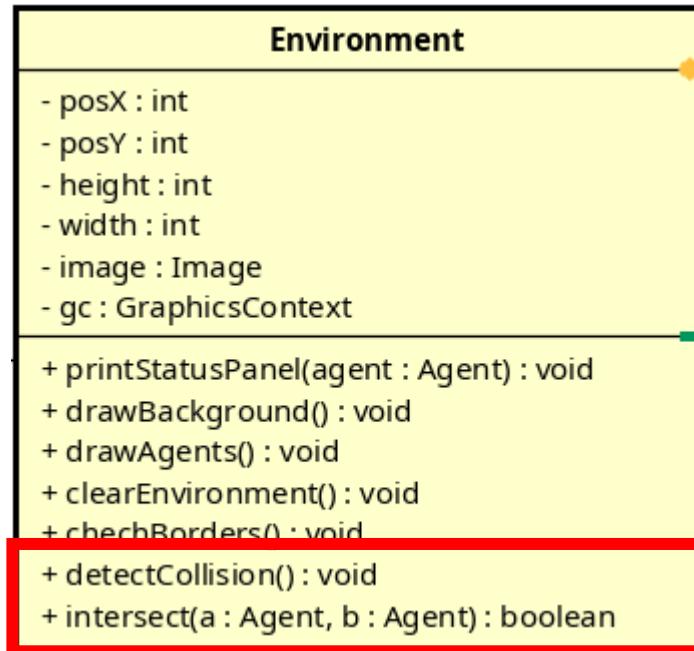
# Axis X & Y: Condition 1, 2, 3 And 4



# Class Environment: New Method



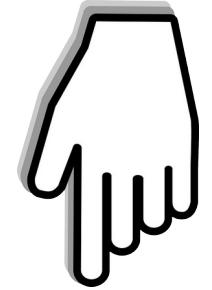
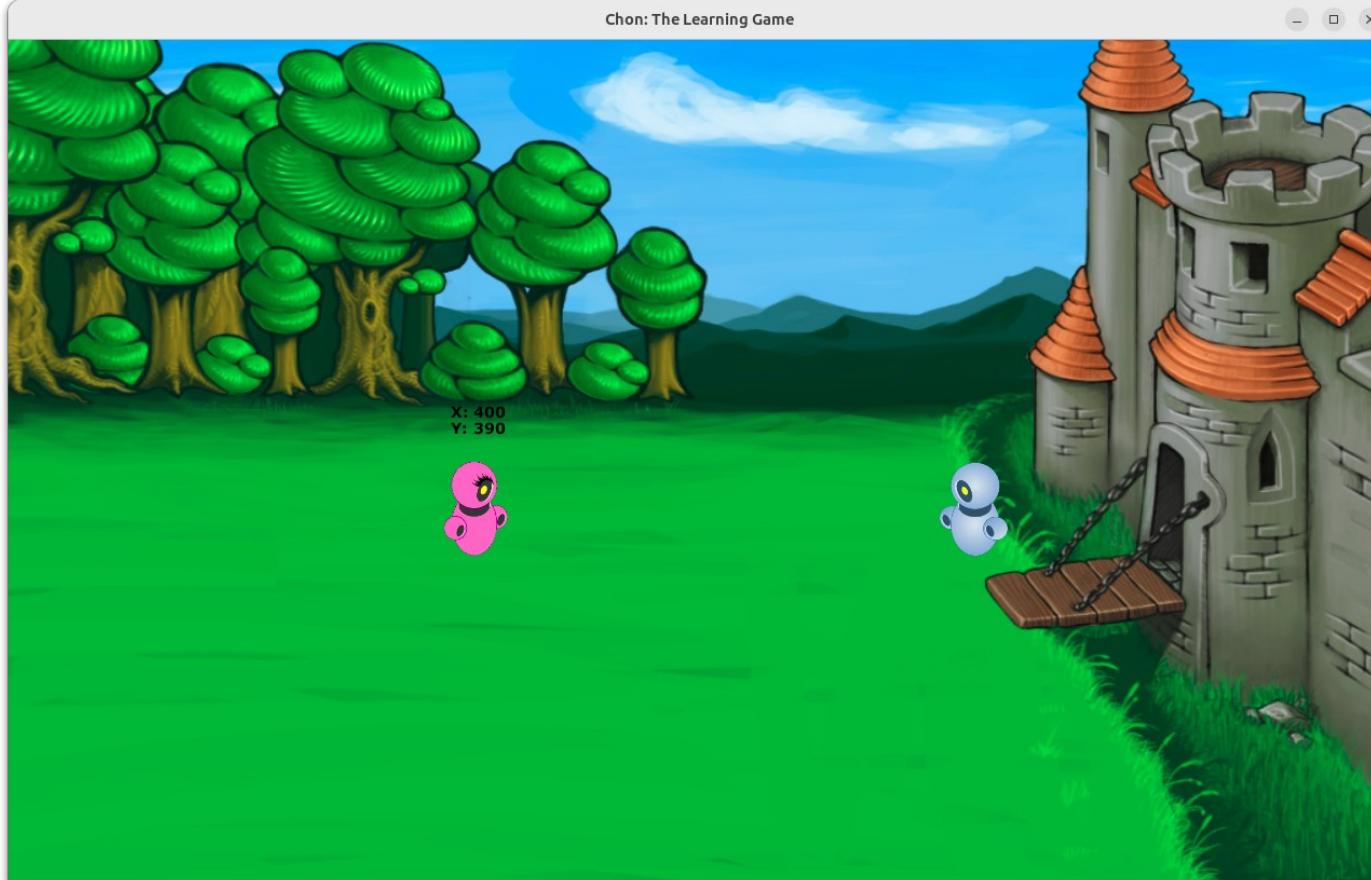
# Class Environment: New Method



# GAME PAUSE



# Pausing the Game



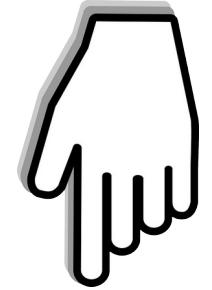
`isPaused = false`

# Pausing the Game



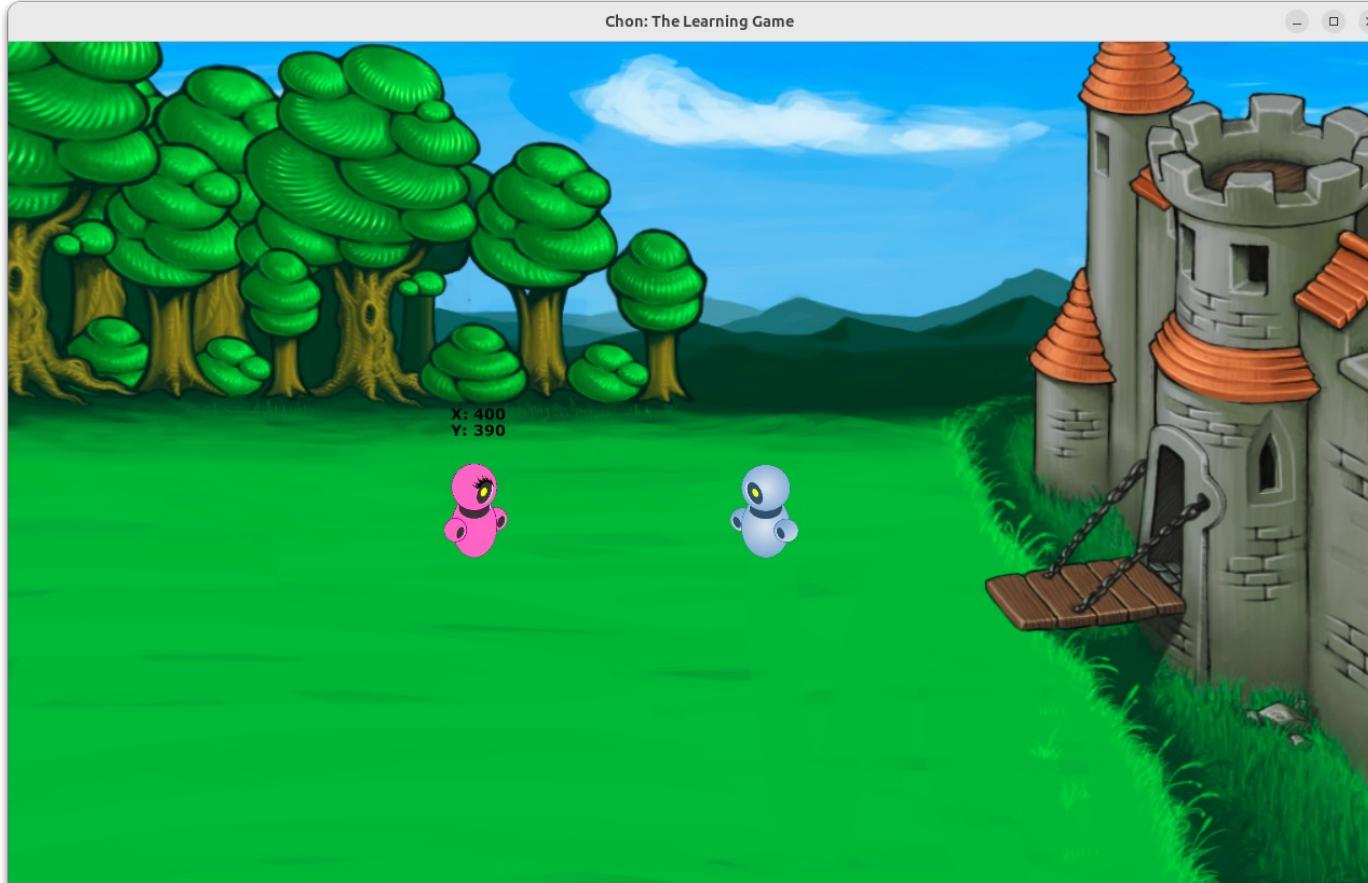
`isPaused = true`

# Pausing the Game



`isPaused = true`

# Pausing the Game

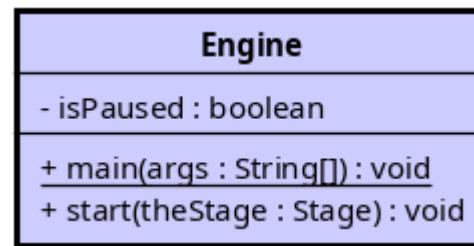


`isPaused = false`

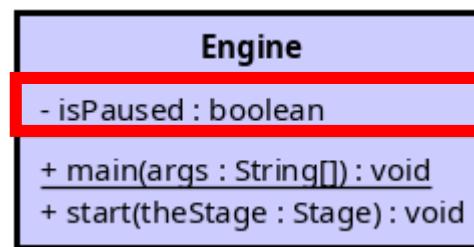
# Branching the Game Loop

```
if (isPaused)
    game paused
    draw pause screen
else
    game dynamics
```

# Class Engine: New Attribute



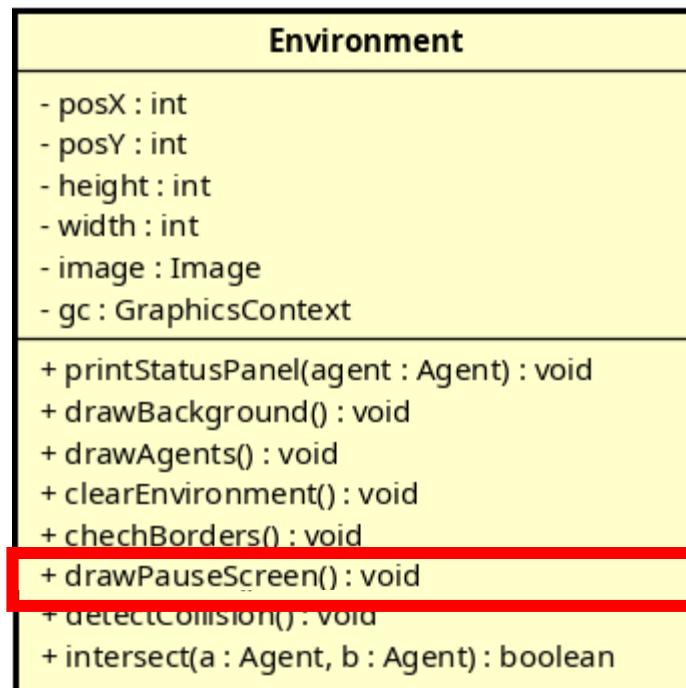
# Class Engine: New Attribute



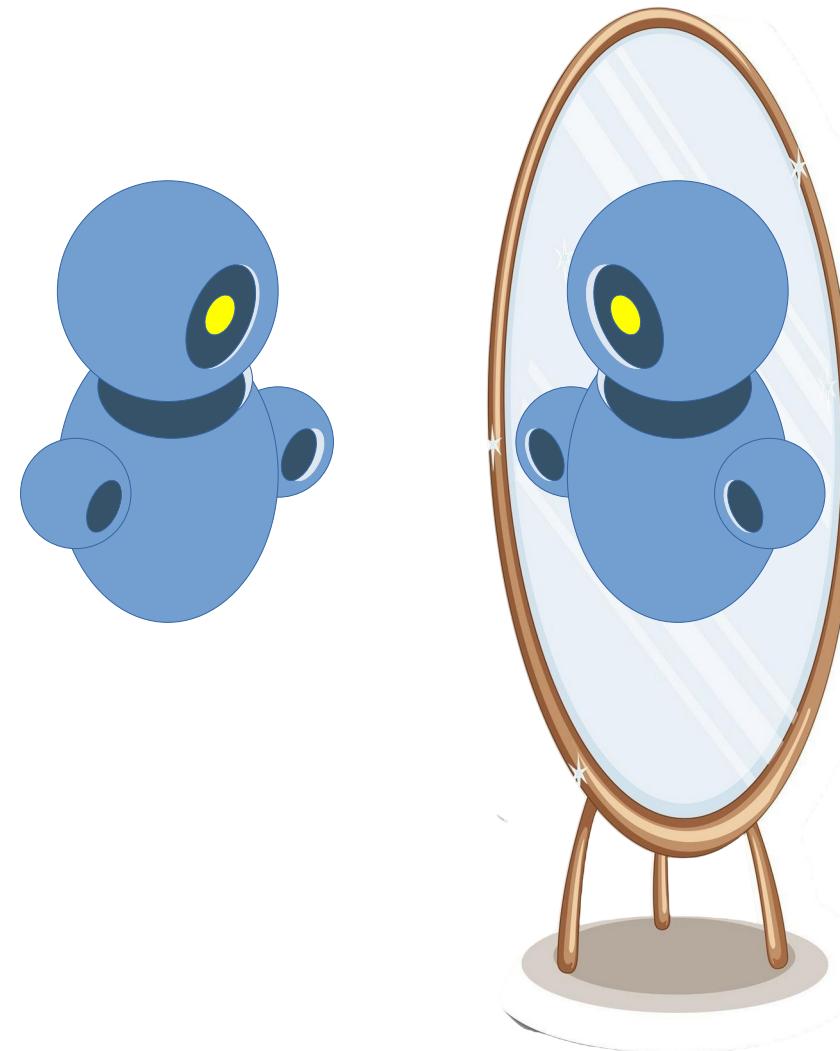
# Class Environment: New Method

Environment
<ul style="list-style-type: none"><li>- posX : int</li><li>- posY : int</li><li>- height : int</li><li>- width : int</li><li>- image : Image</li><li>- gc : GraphicsContext</li></ul>
<ul style="list-style-type: none"><li>+ printStatusPanel(agent : Agent) : void</li><li>+ drawBackground() : void</li><li>+ drawAgents() : void</li><li>+ clearEnvironment() : void</li><li>+ chechBorders() : void</li><li>+ drawPauseScreen() : void</li><li>+ detectCollision() : void</li><li>+ intersect(a : Agent, b : Agent) : boolean</li></ul>

# Class Environment: New Method



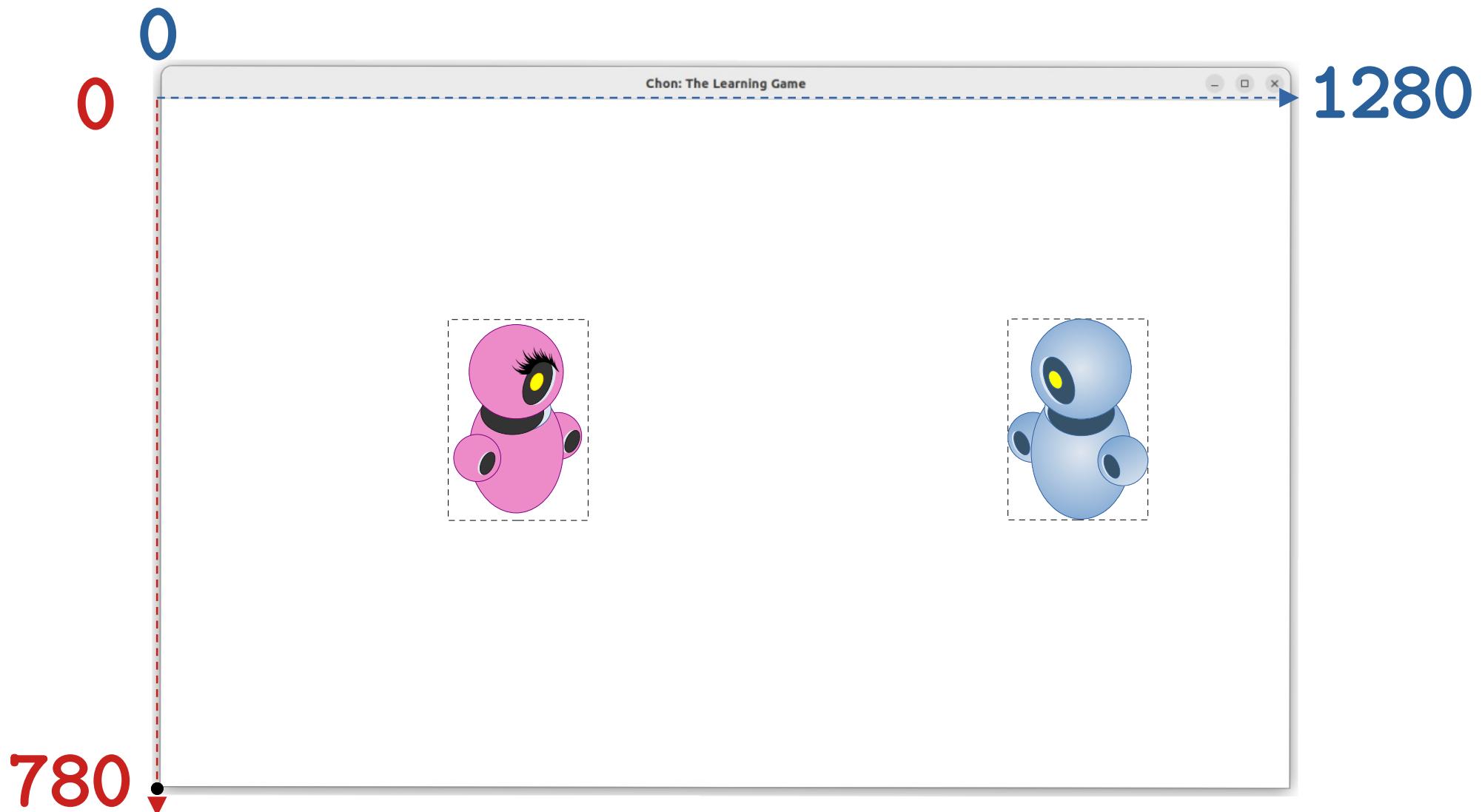
# FLIP IMAGE



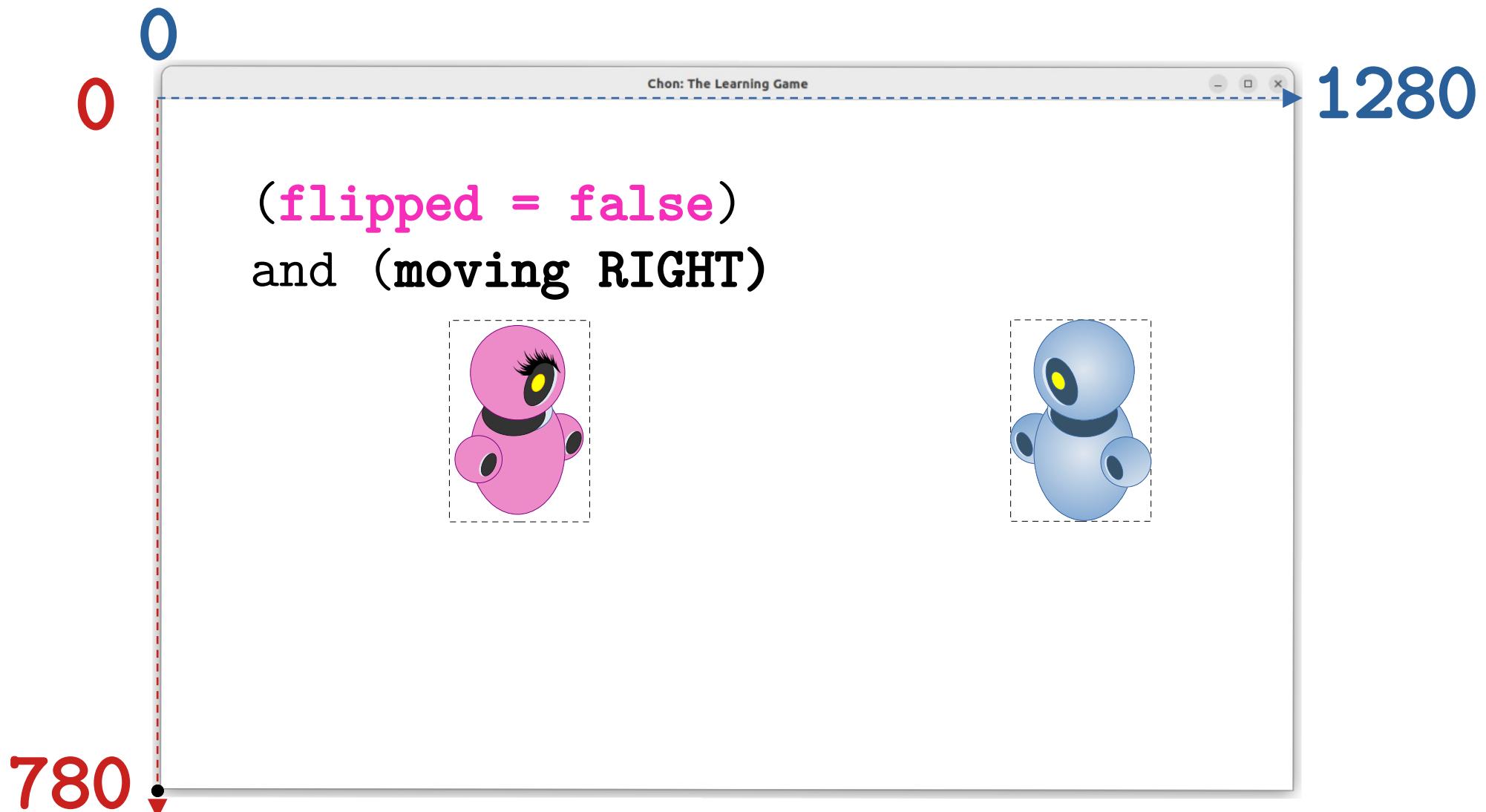
# Flipped Image

A **flipped** image is a **mirrored** or **reversed** image along the horizontal axis.

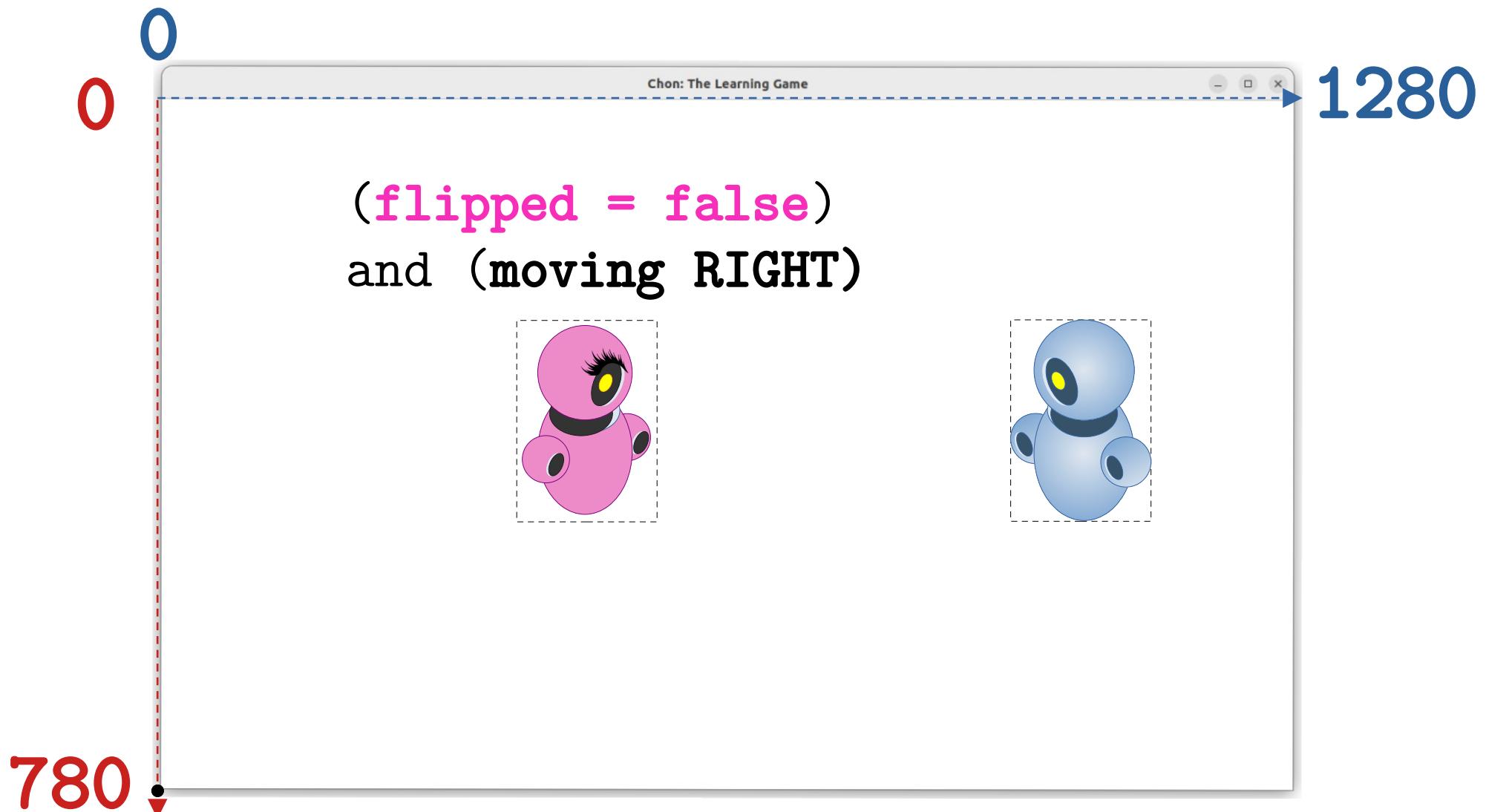
# The Initial Position of an Image



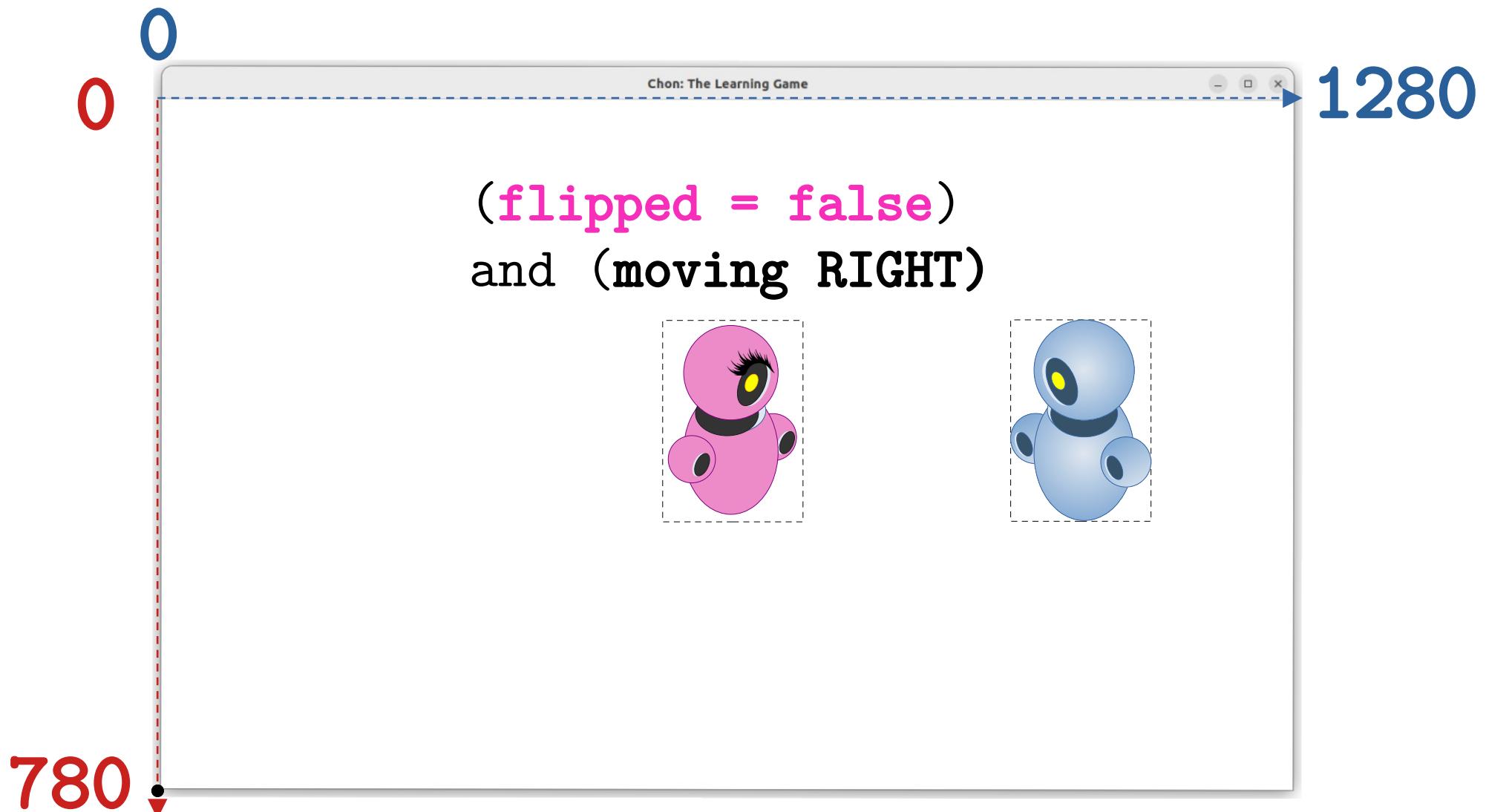
# Moving Right: Case 1



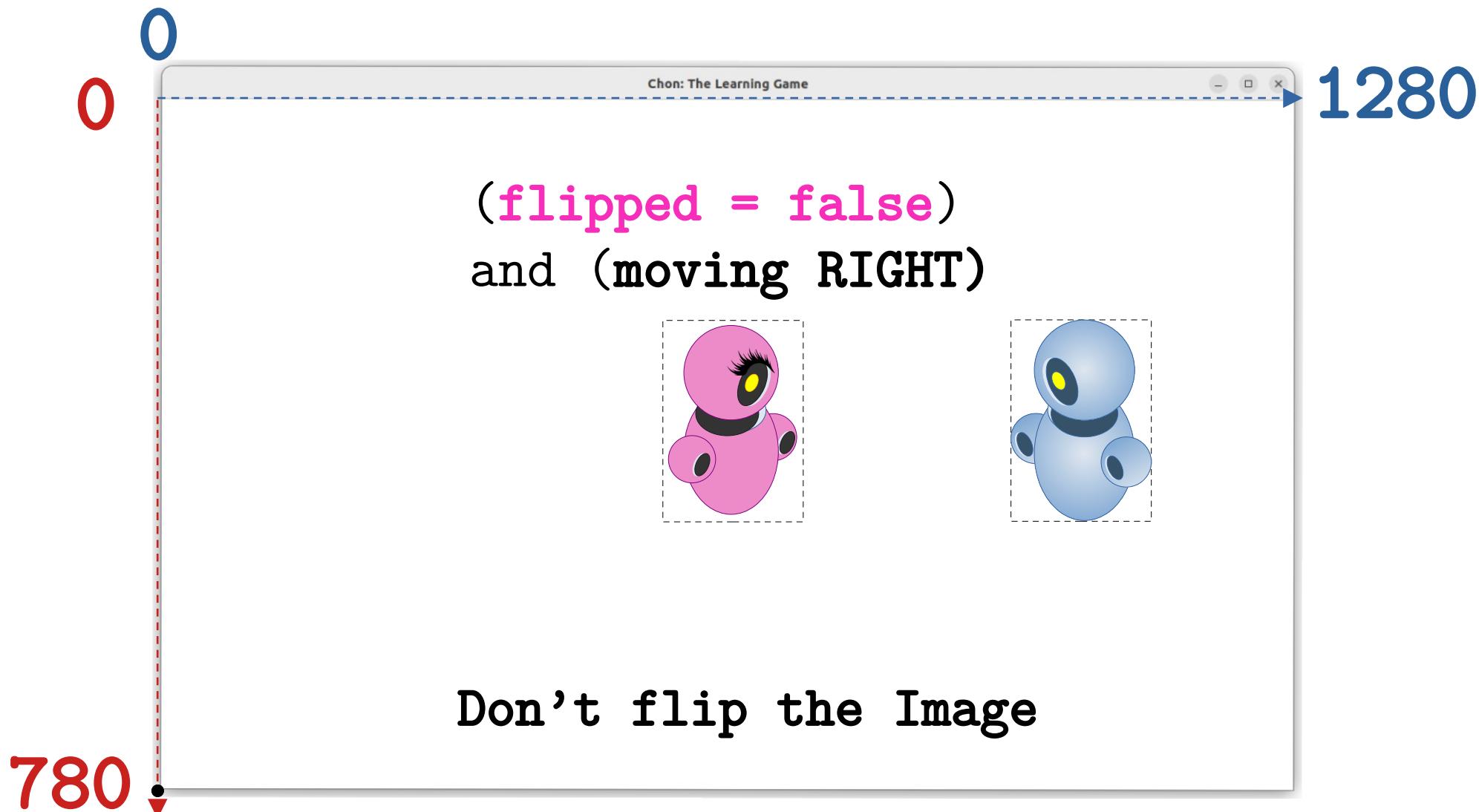
# Moving Right: Case 1



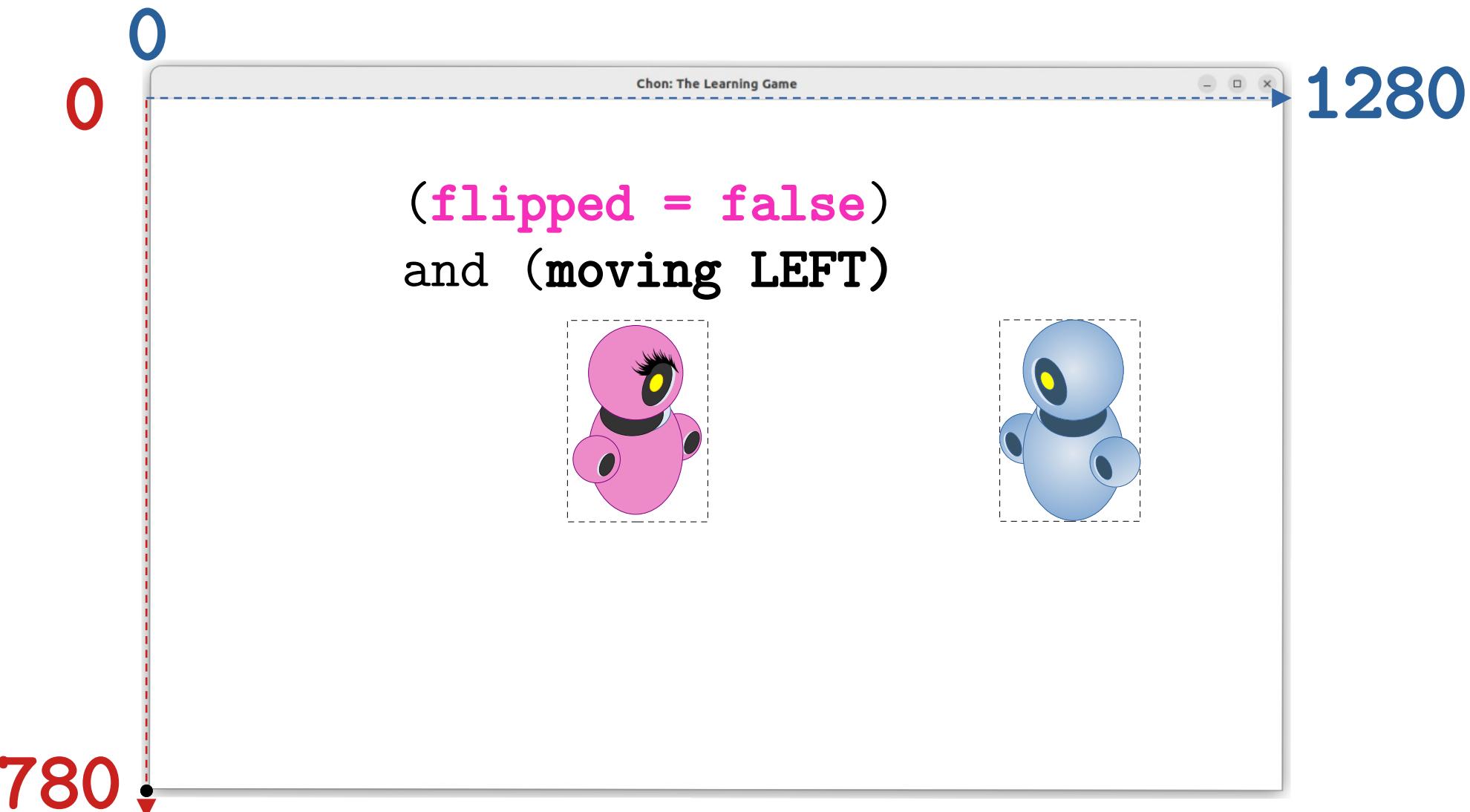
# Moving Right: Case 1



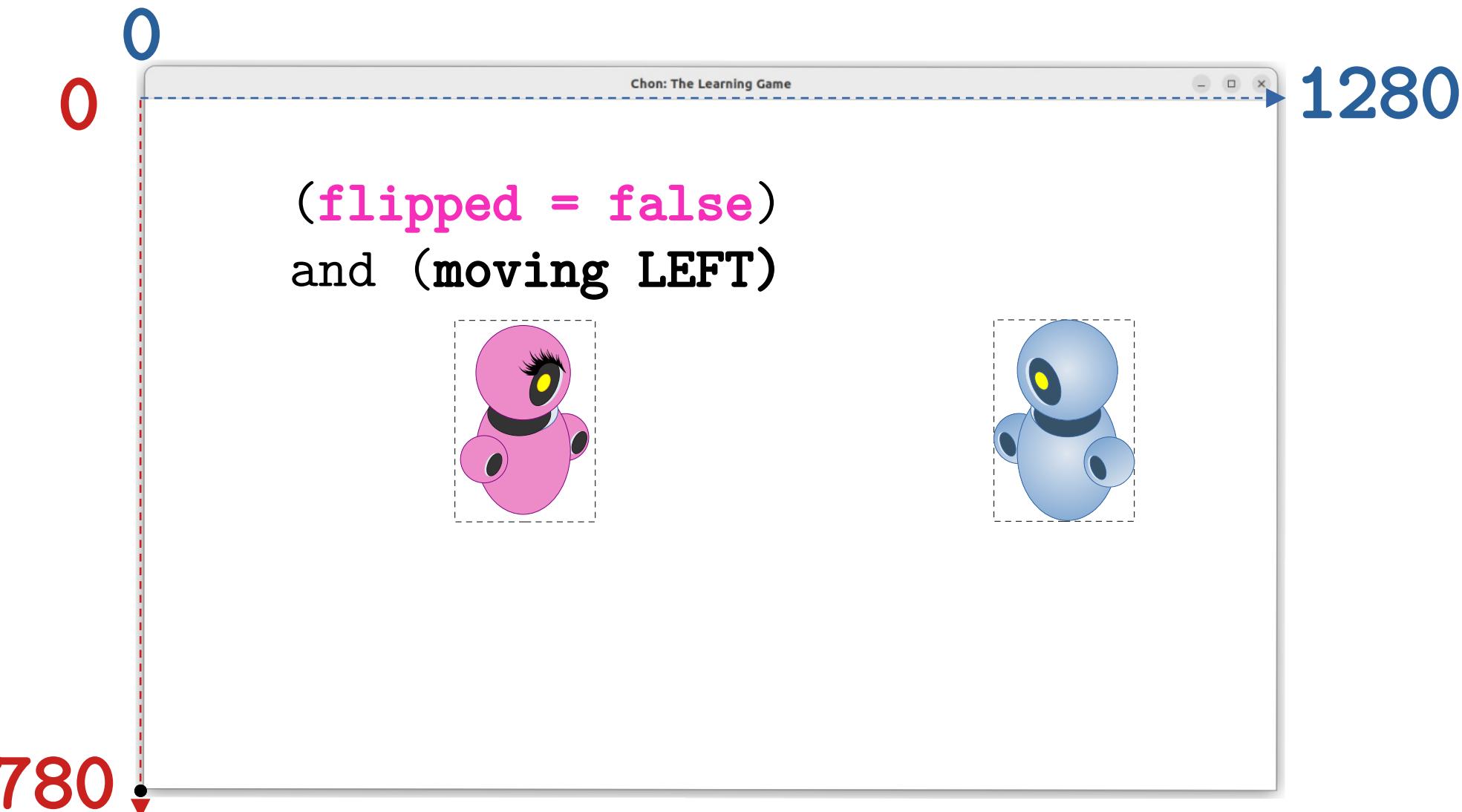
# Moving Right: Case 1



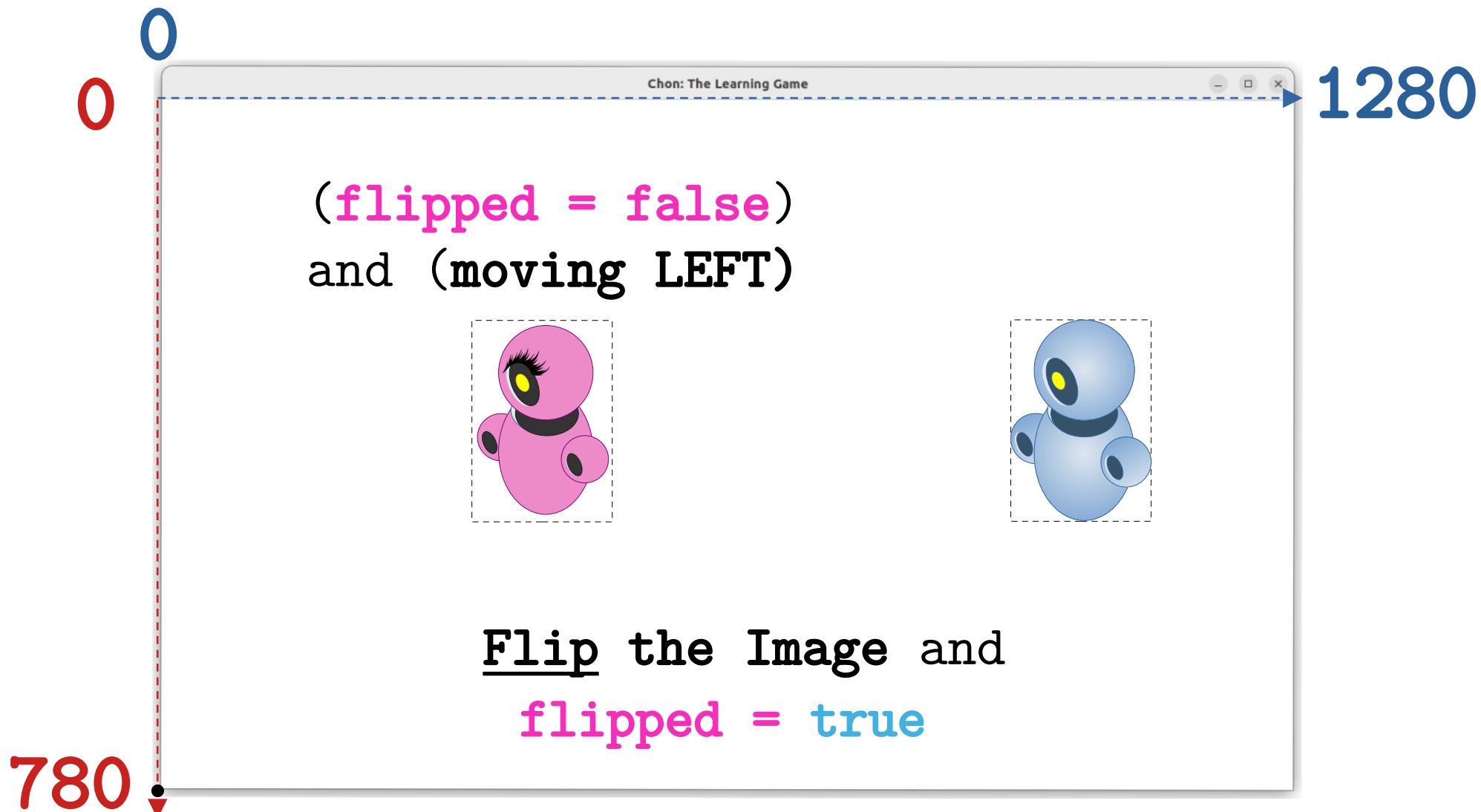
# Moving Left: Case 2



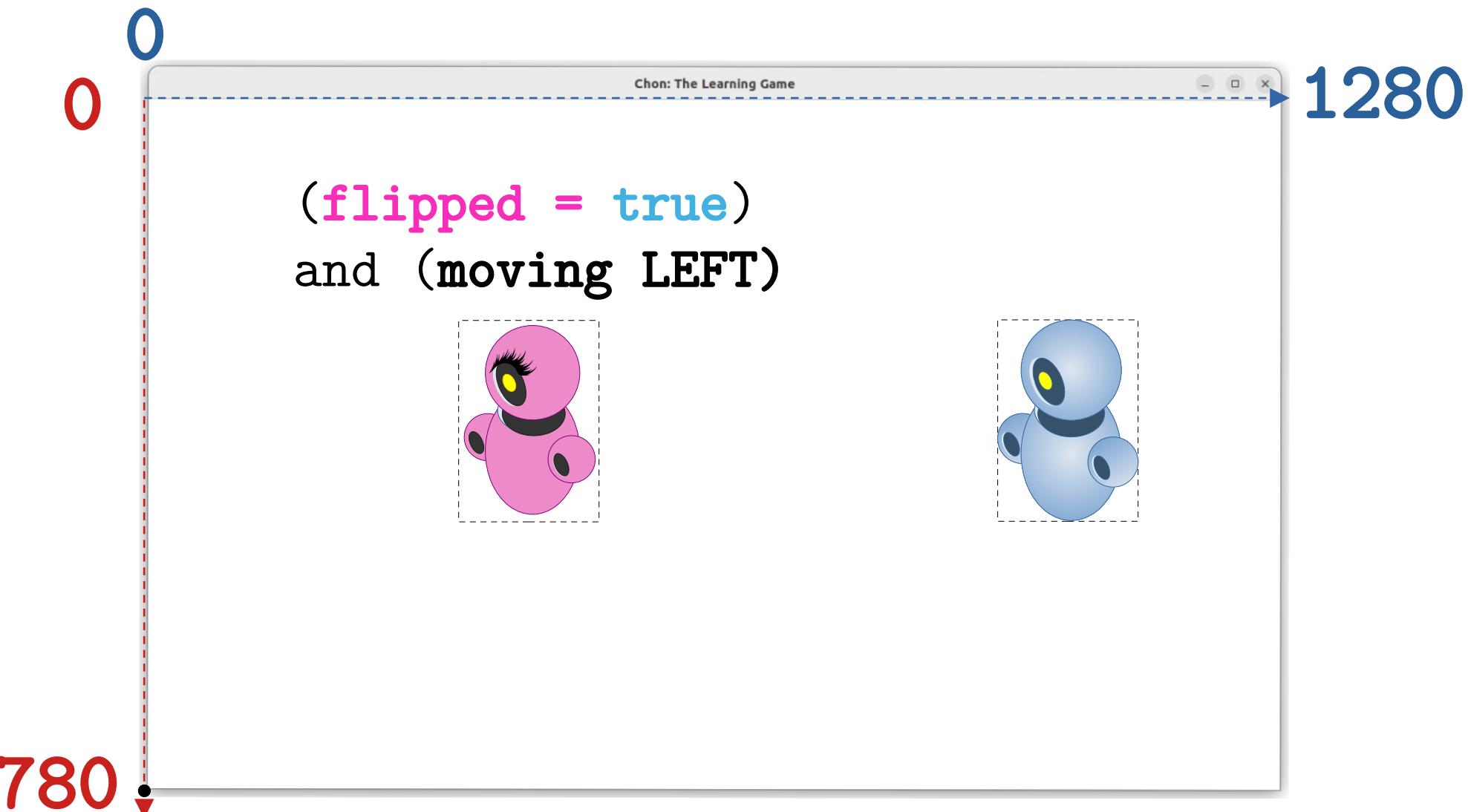
# Moving Left: Case 2



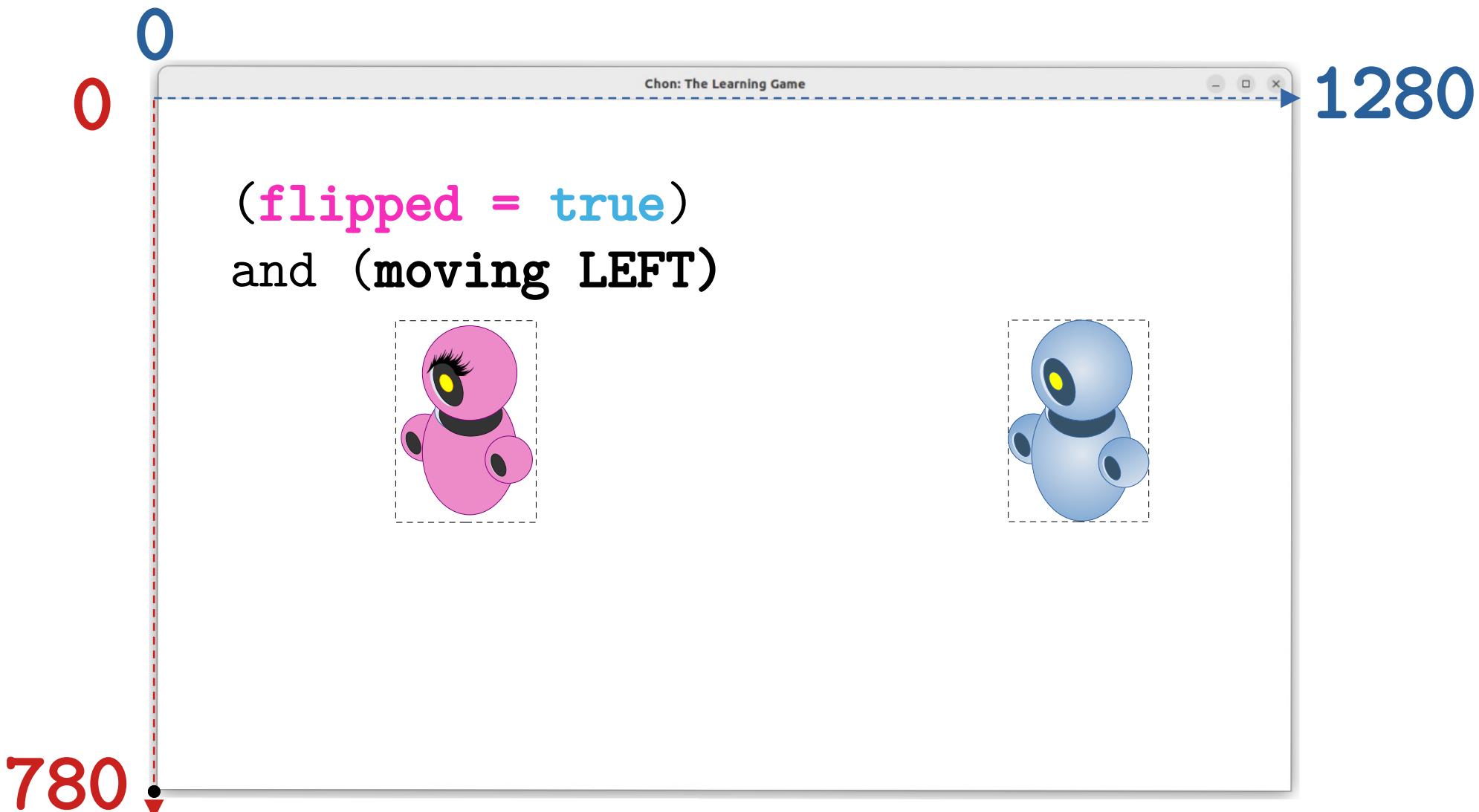
# Moving Left: Case 2



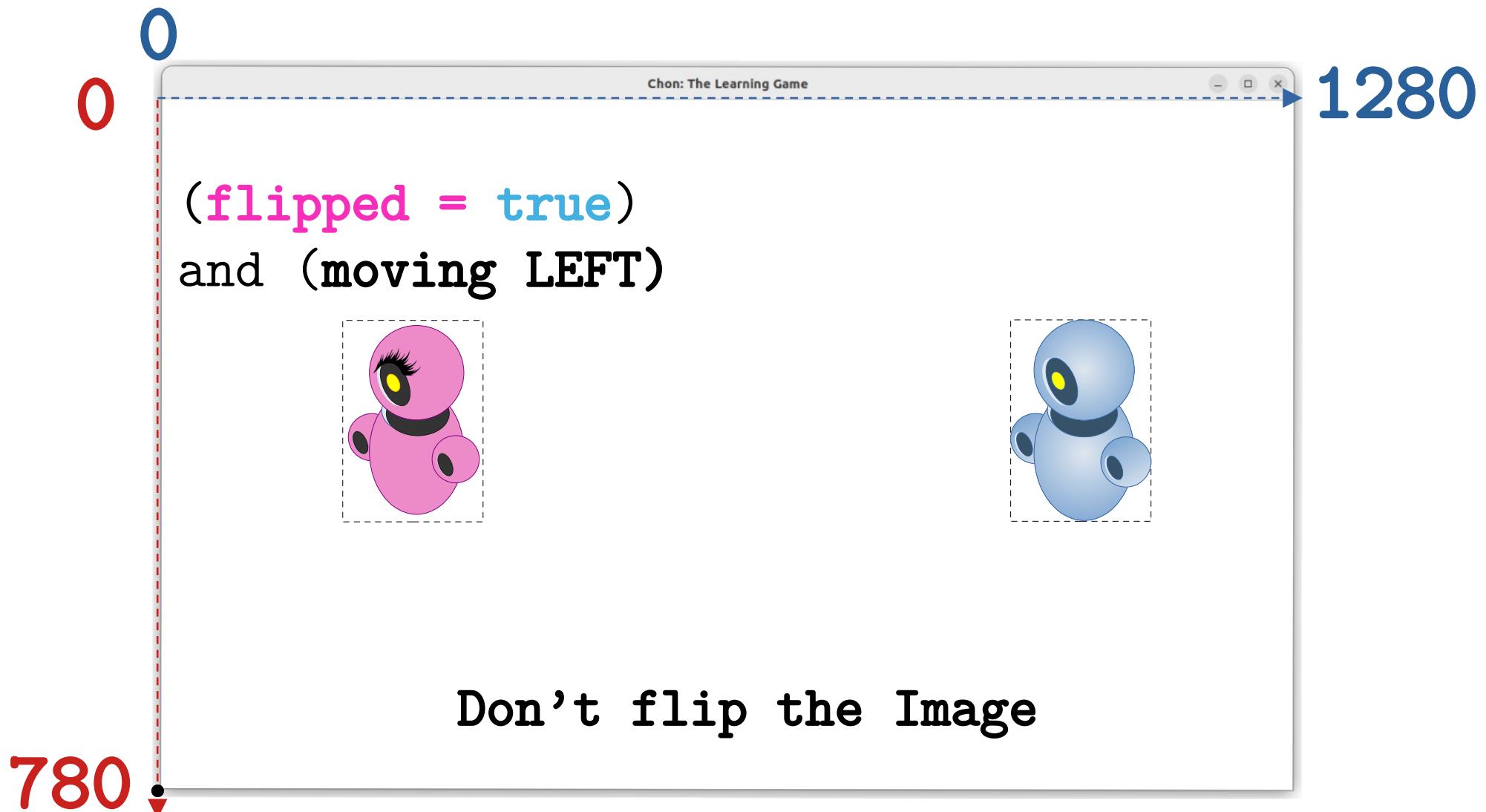
# Moving Left: Case 3



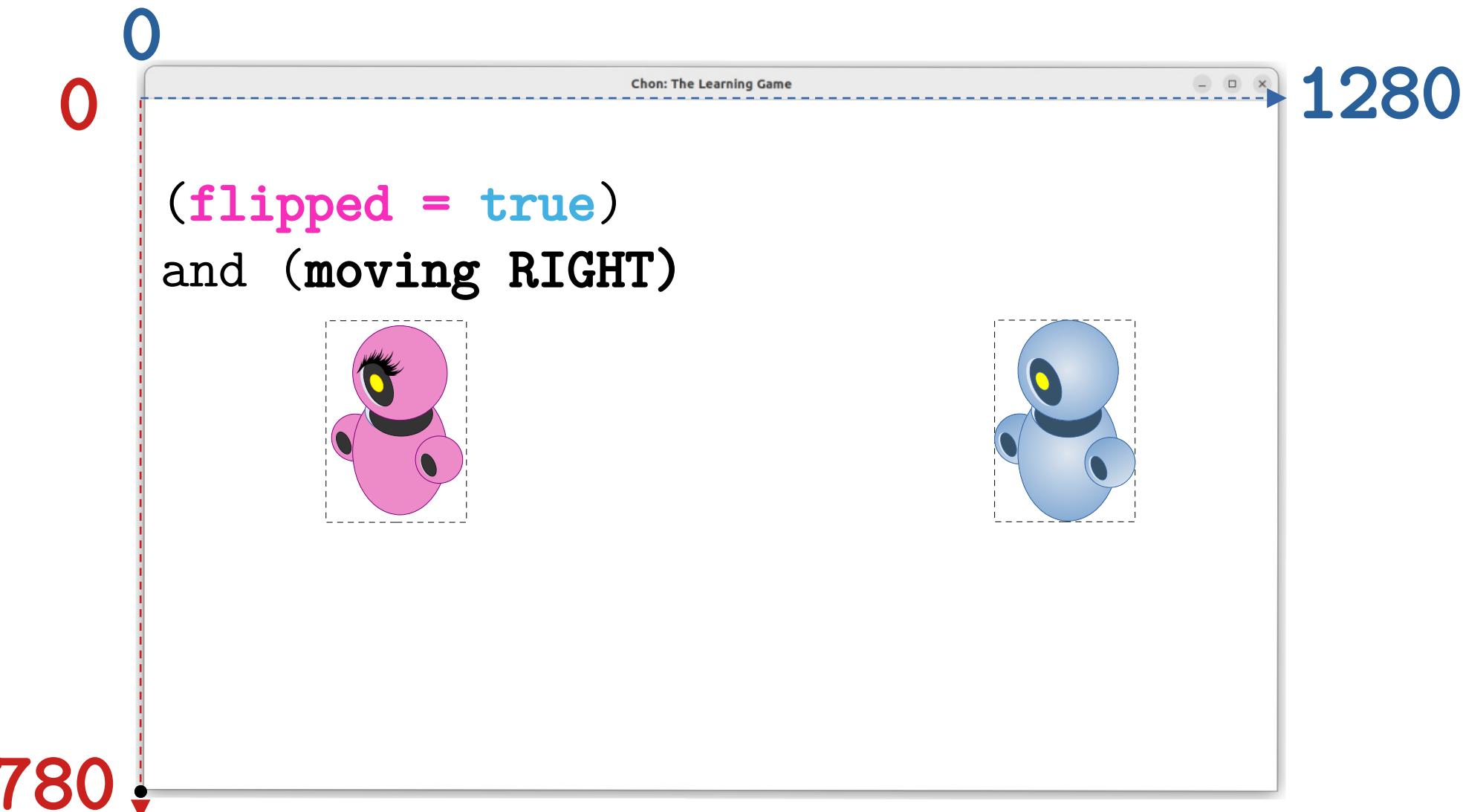
# Moving Left: Case 3



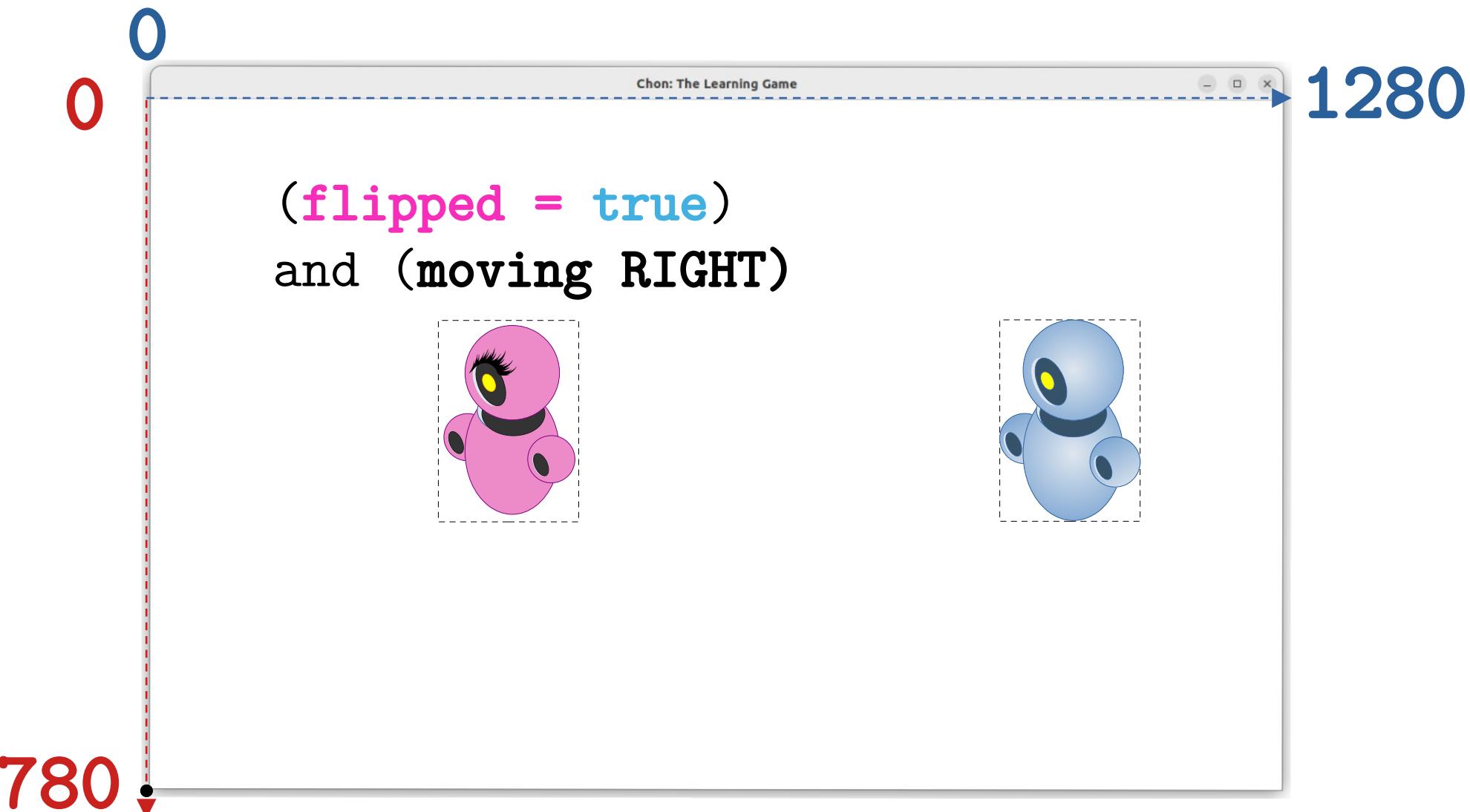
# Moving Left: Case 3



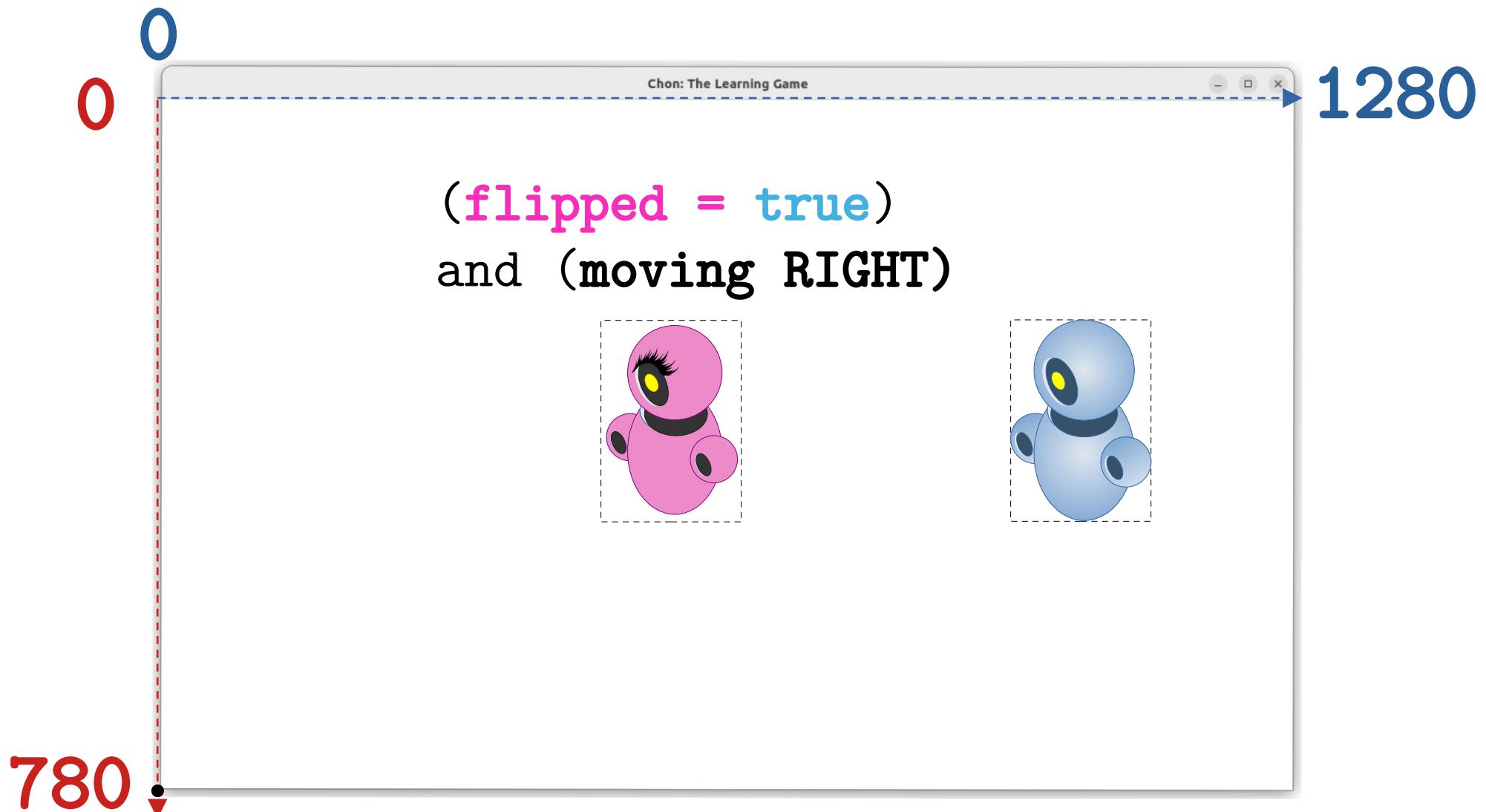
# Moving Left: Case 4



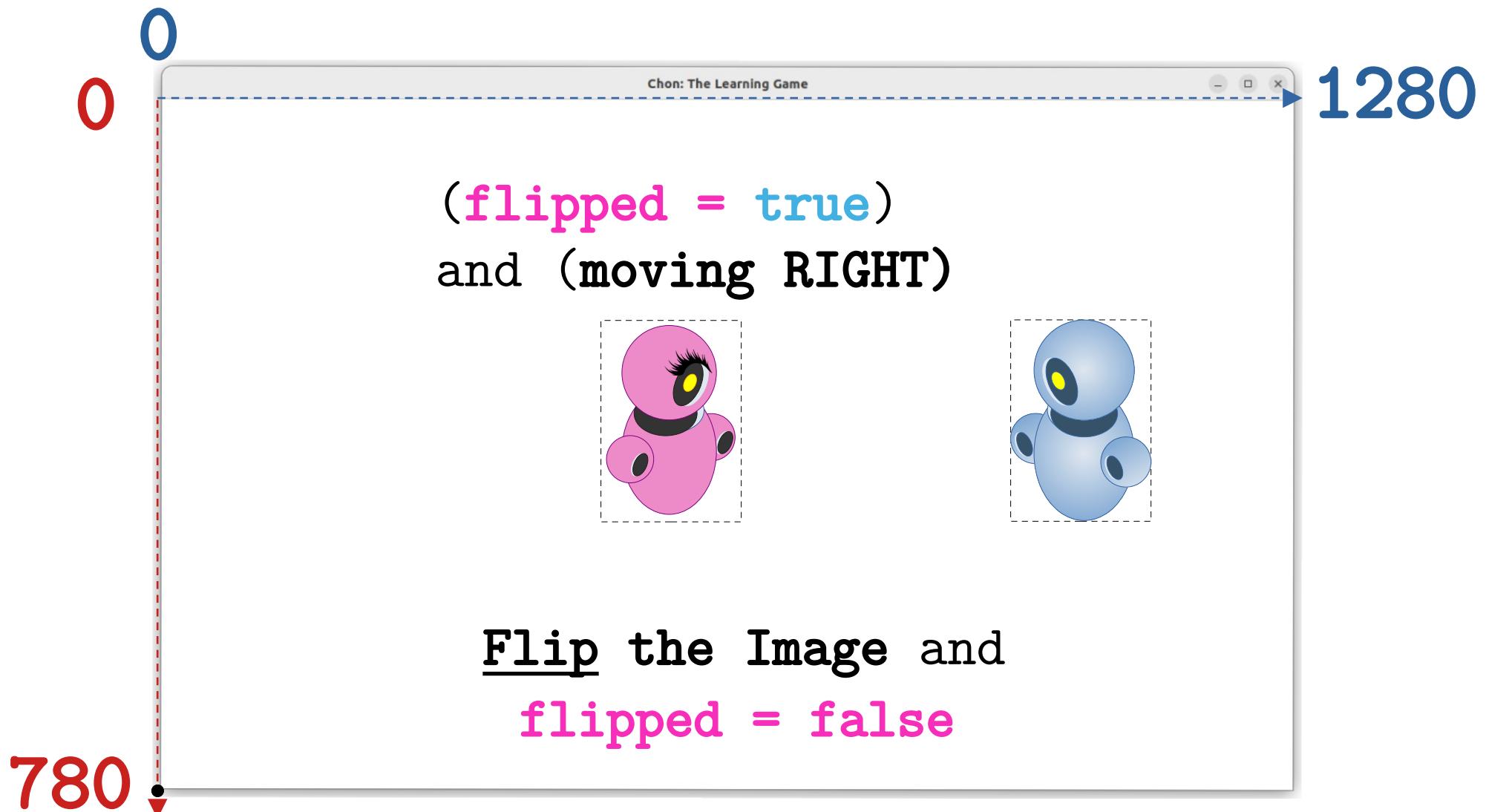
# Moving Left: Case 4



# Moving Left: Case 4



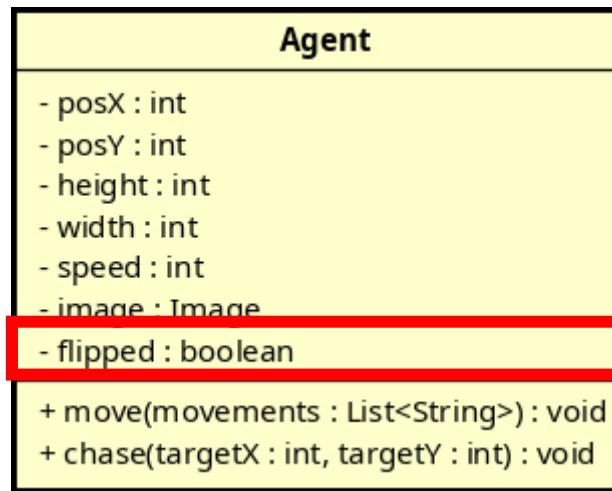
# Moving Left: Case 4



# Class Agent: New Attribute

Agent
- posX : int
- posY : int
- height : int
- width : int
- speed : int
- image : Image
- flipped : boolean
+ move(movements : List<String>) : void
+ chase(targetX : int, targetY : int) : void

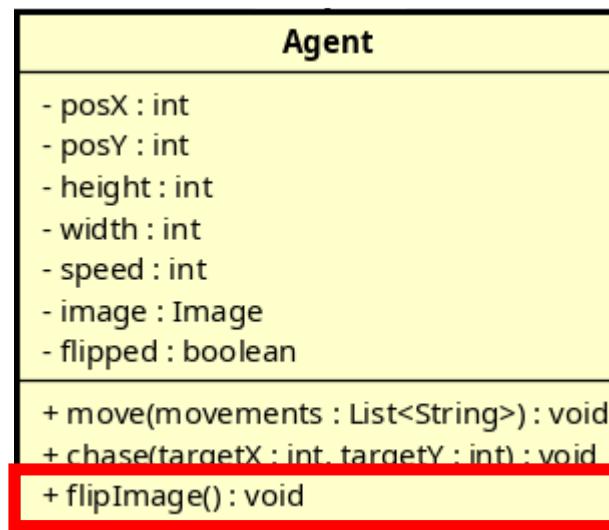
# Class Agent: New Attribute



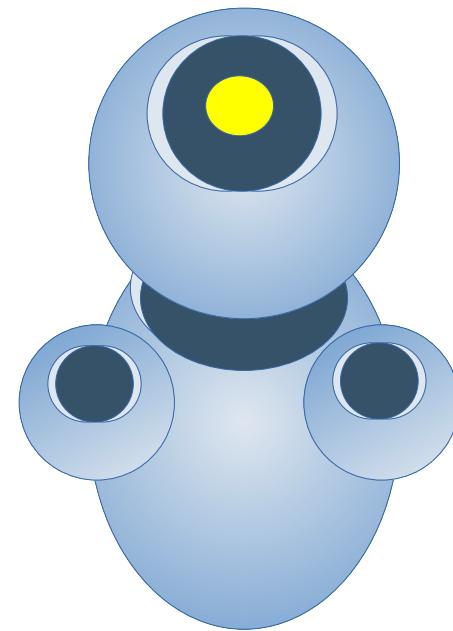
# Class Agent: New Method

Agent	
- posX : int	
- posY : int	
- height : int	
- width : int	
- speed : int	
- image : Image	
- flipped : boolean	
+ move(movements : List<String>) : void	
+ chase(targetX : int, targetY : int) : void	
+ flipImage() : void	

# Class Agent: New Method



# LIFE BAR



# The Life Bar



# The Life Bar: Height

barHeight



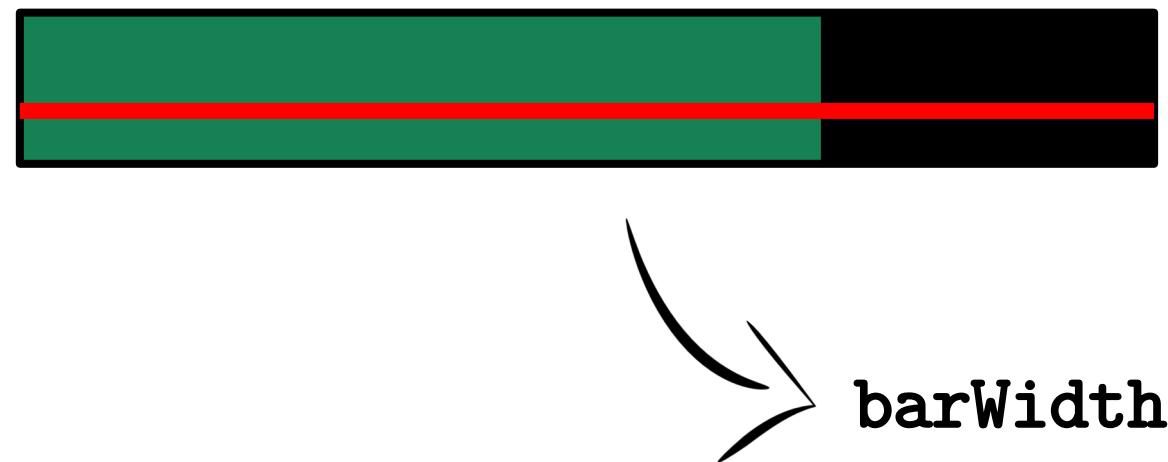
# The Life Bar: Height



# The Life Bar: Width



# The Life Bar: Width



# The Life Bar: Width



**barWidth = agent's width**

# The Life Bar: Border Thickness



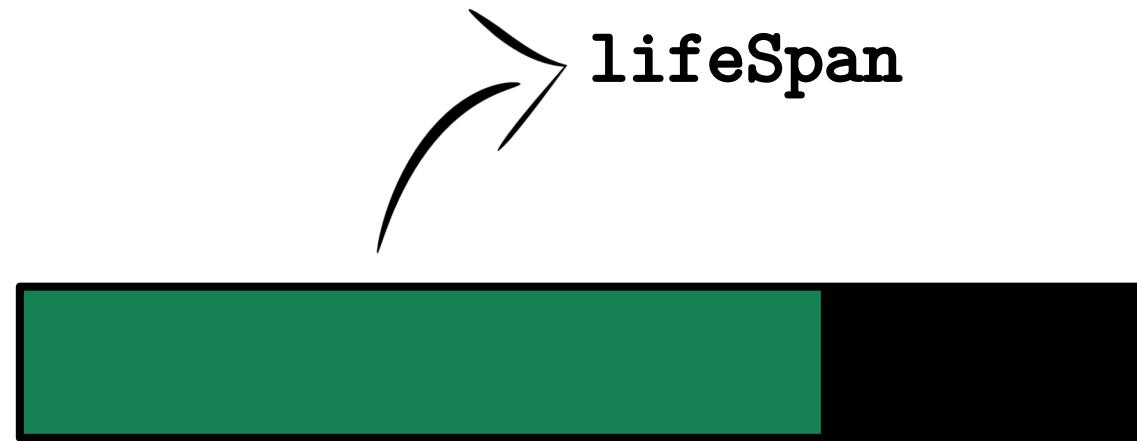
`borderThickness`

# The Life Bar: Border Thickness

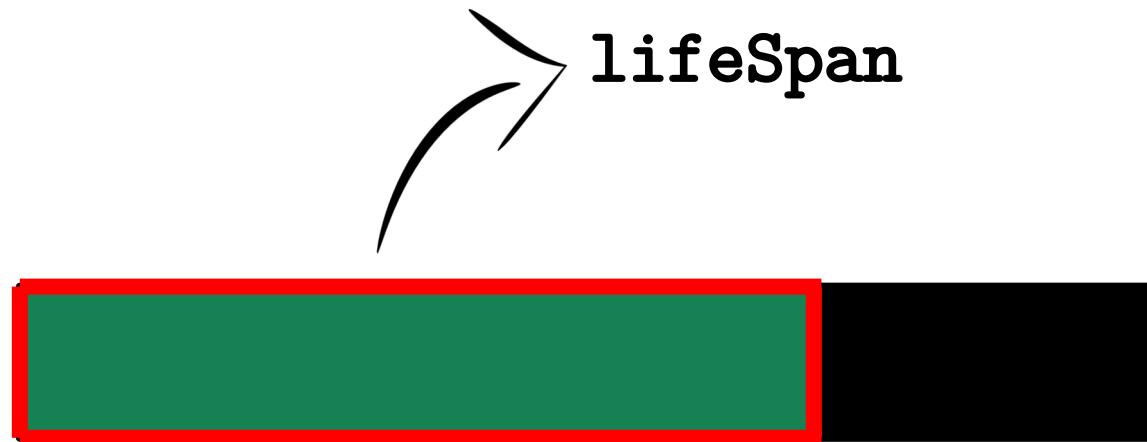


`borderThickness`

# The Life Bar: Life Span



# The Life Bar: Life Span



# The Life Bar: Life Span



# The Life Bar: Life Span



**life percentage = actual health / maximum health**

# The Life Bar: Life Span



`life percentage = actual health / maximum health`

`life span = (life percentage * agent's width) / 100`

# The Life Bar: Life Span



life percentage = actual health / maximum health

$$80 \text{ / } 100 = 0,8$$

life span = (life percentage \* agent's width) / 100

# The Life Bar: Life Span



life percentage = actual health / maximum health

$$80 / 100 = 0,8$$

life span = (life percentage \* agent's width) / 100

$$0,8 * 65 = 52$$

# The Life Bar: Life Span



life percentage = actual health / maximum health

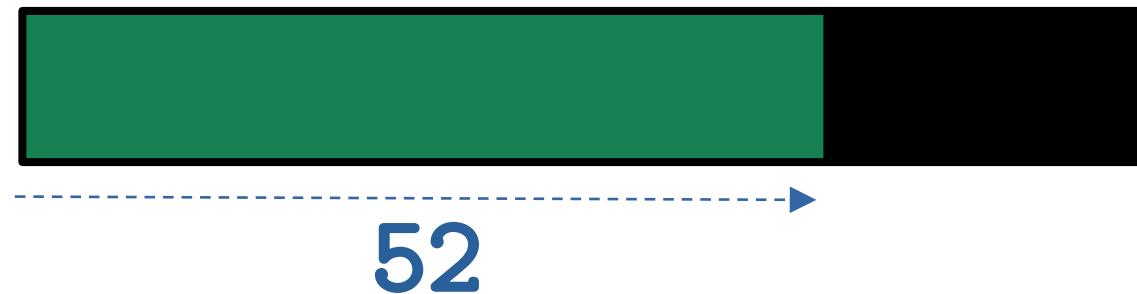
$$80 / 100 = 0,8$$

life span = (life percentage \* agent's width) / 100

$$0,8 * 65 = 52$$

52

# The Life Bar: Life Span



life percentage = actual health / maximum health

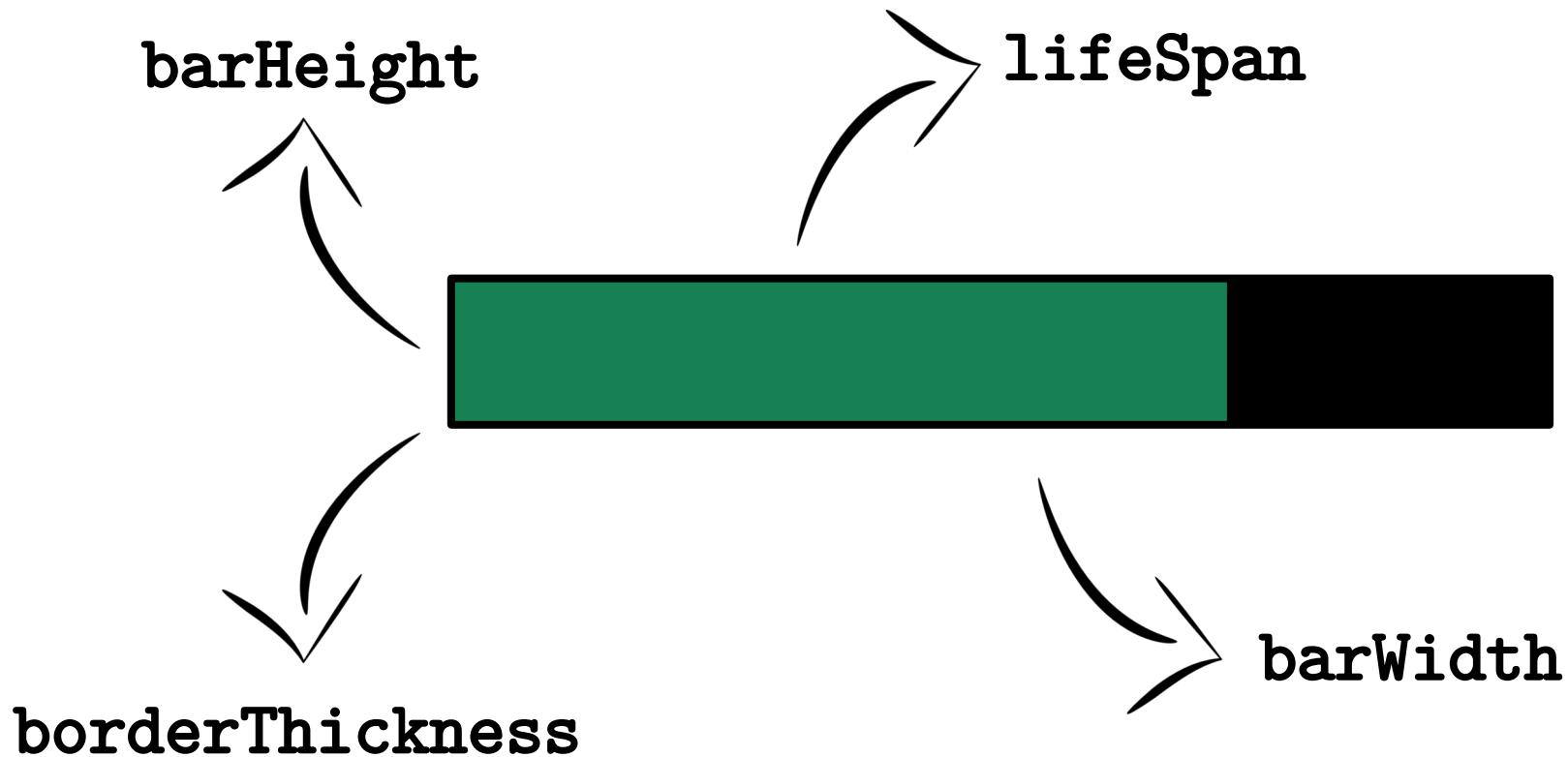
$$80 / 100 = 0,8$$

life span = (life percentage \* agent's width) / 100

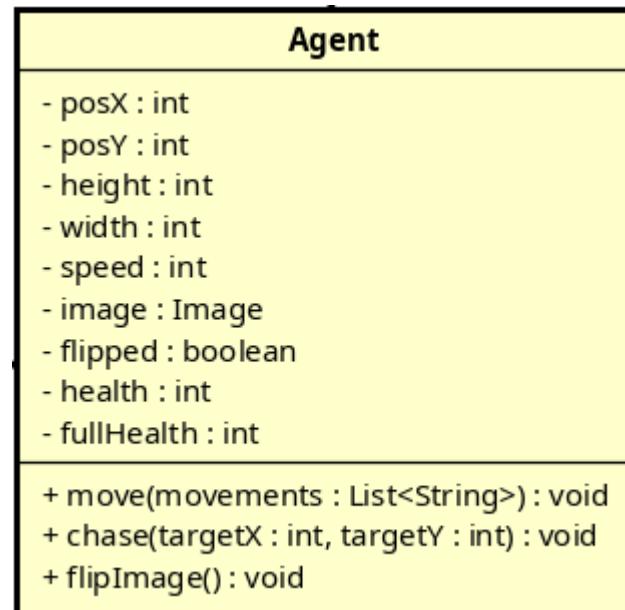
$$0,8 * 65 = 52$$

52

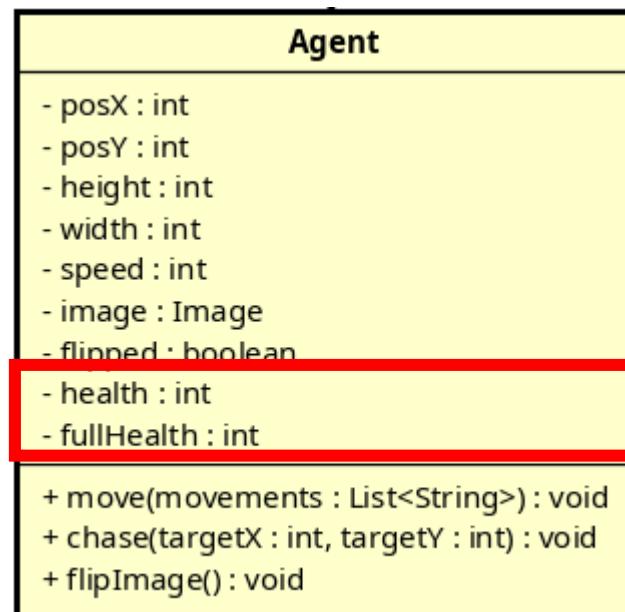
# The Life Bar



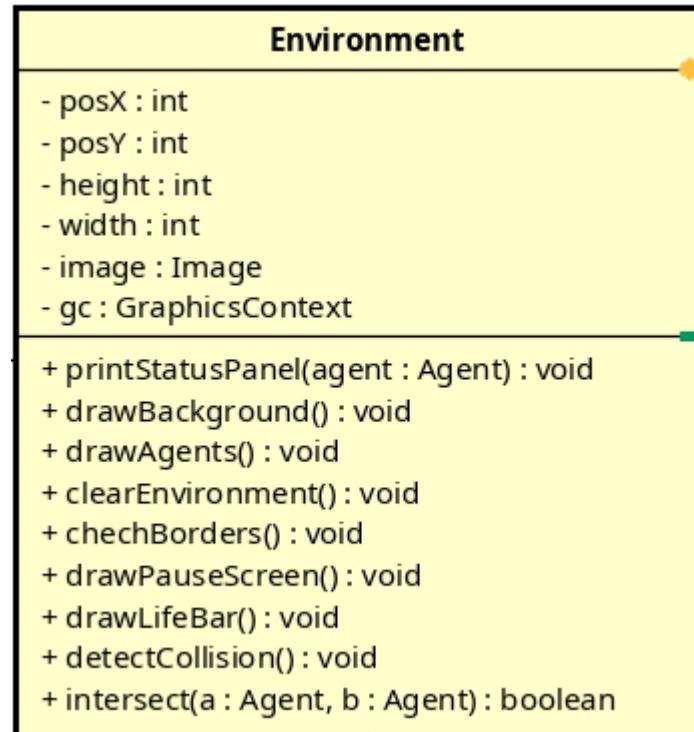
# Class Agent: New Attributes



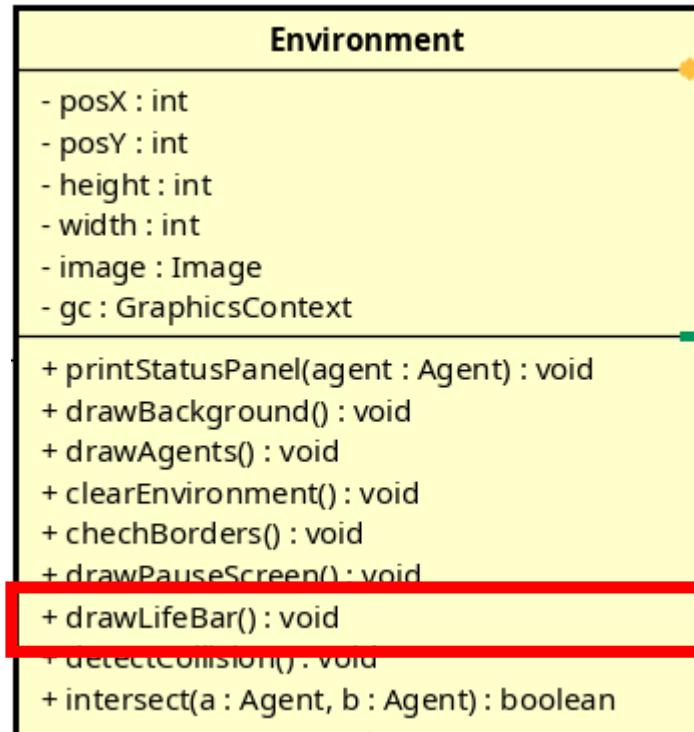
# Class Agent: New Attributes



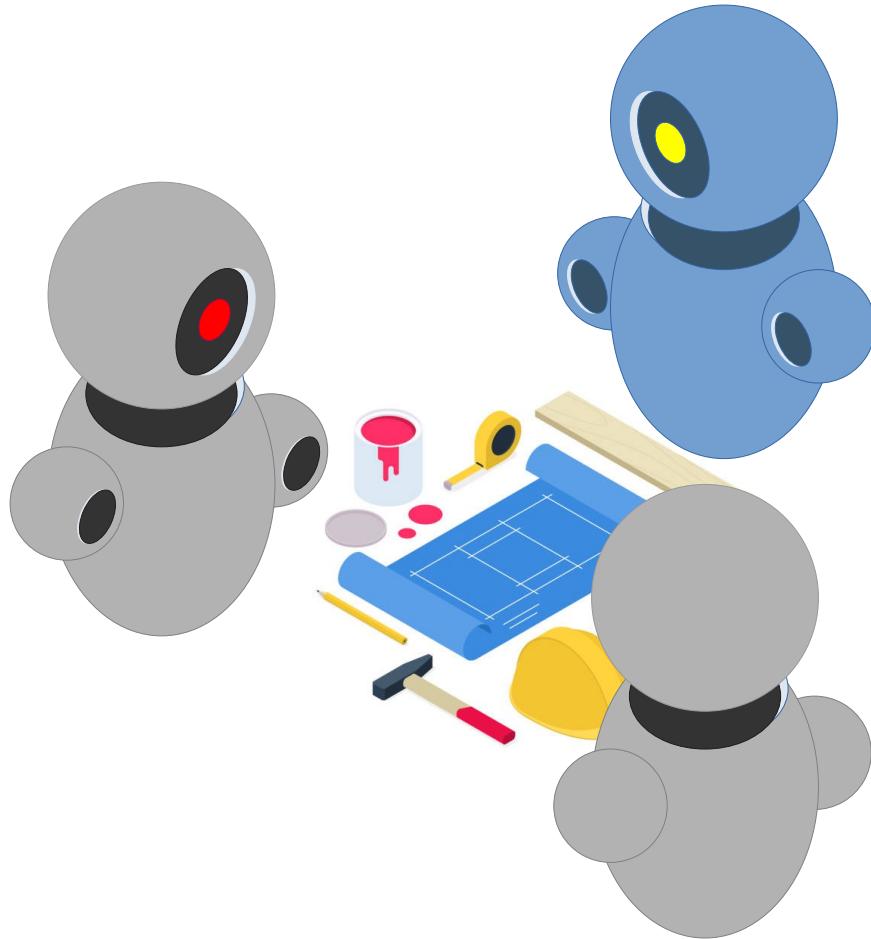
# Class Environment: New Method



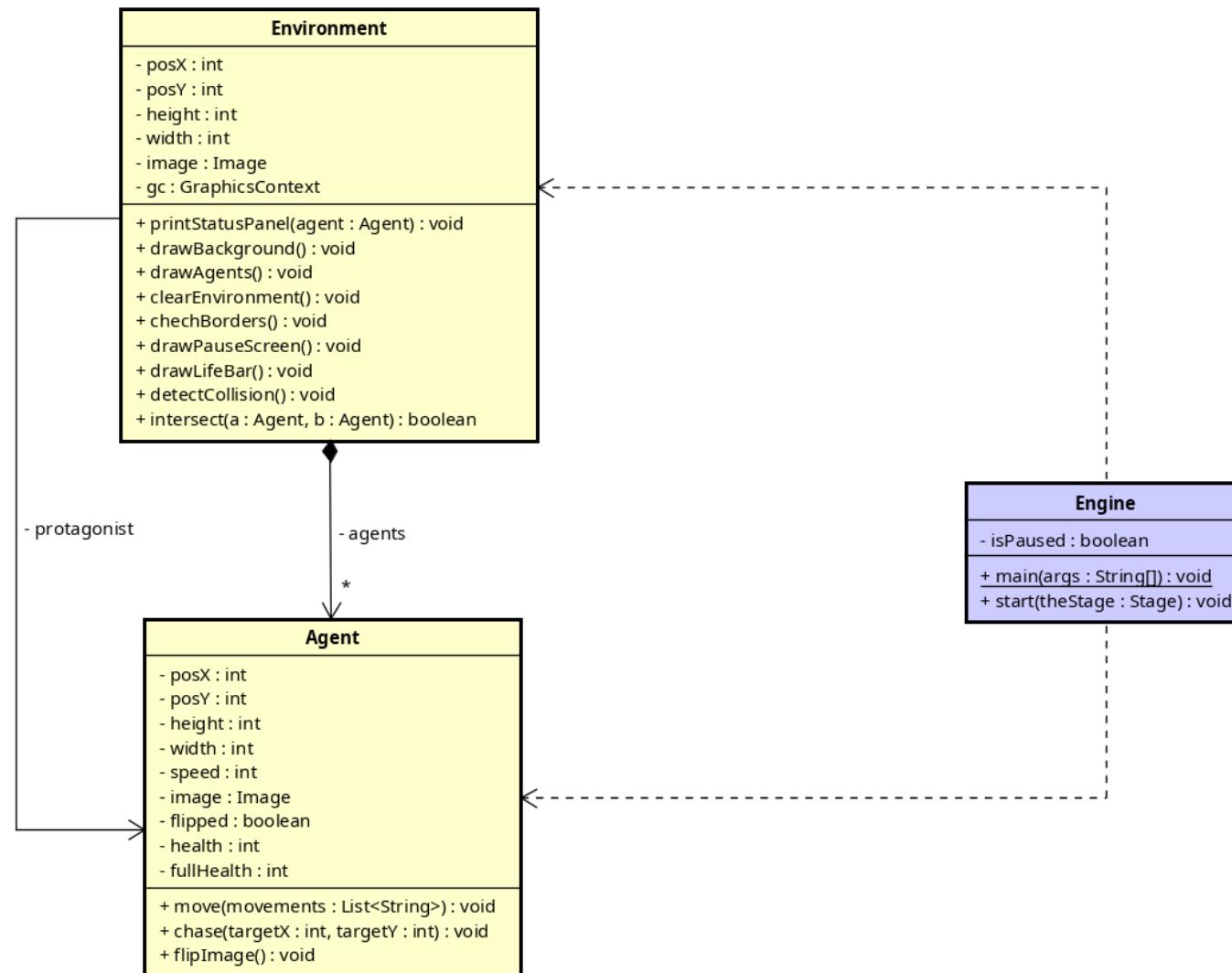
# Class Environment: New Method



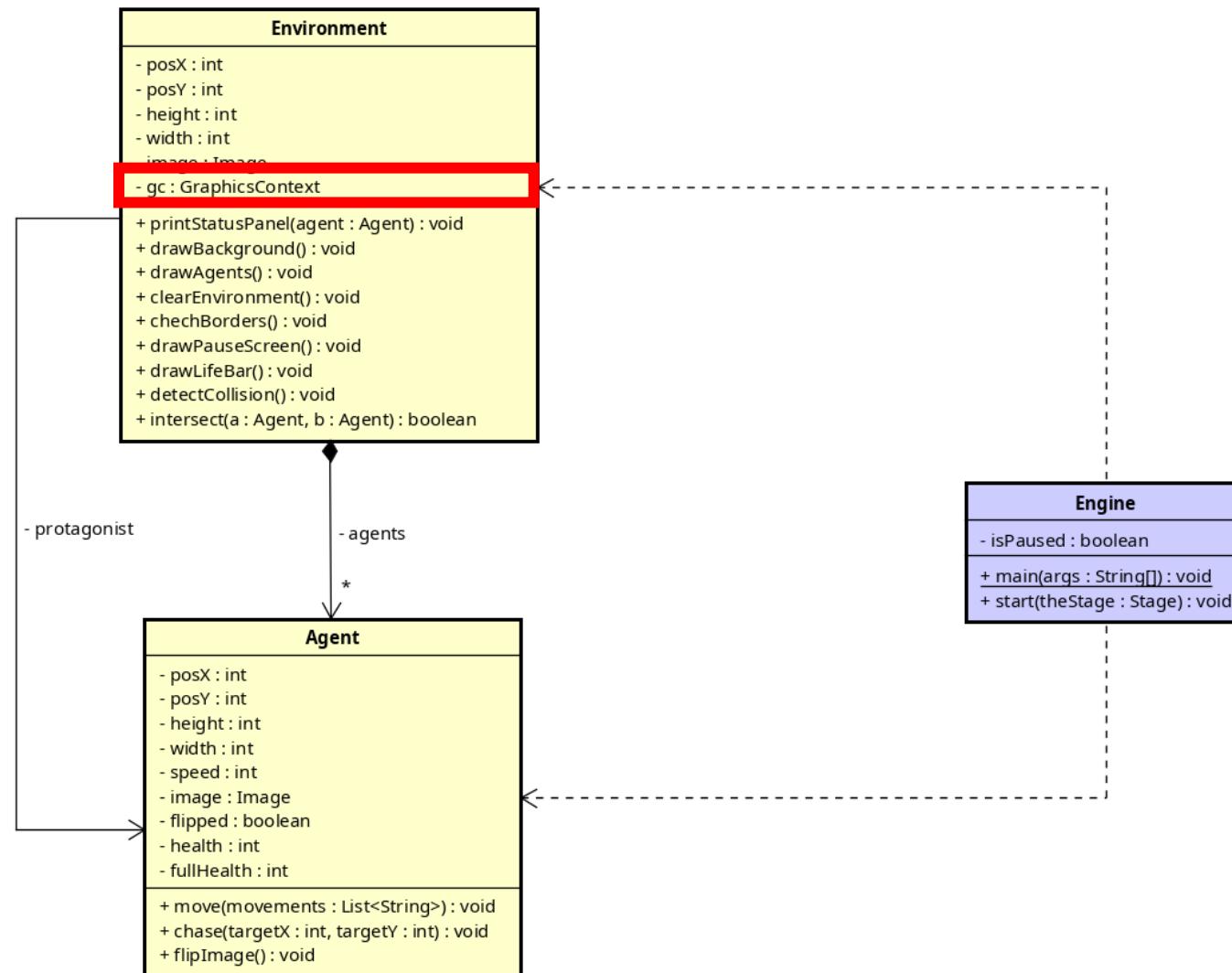
# RENDERING THE GAME: THE MEDIATOR PATTERN



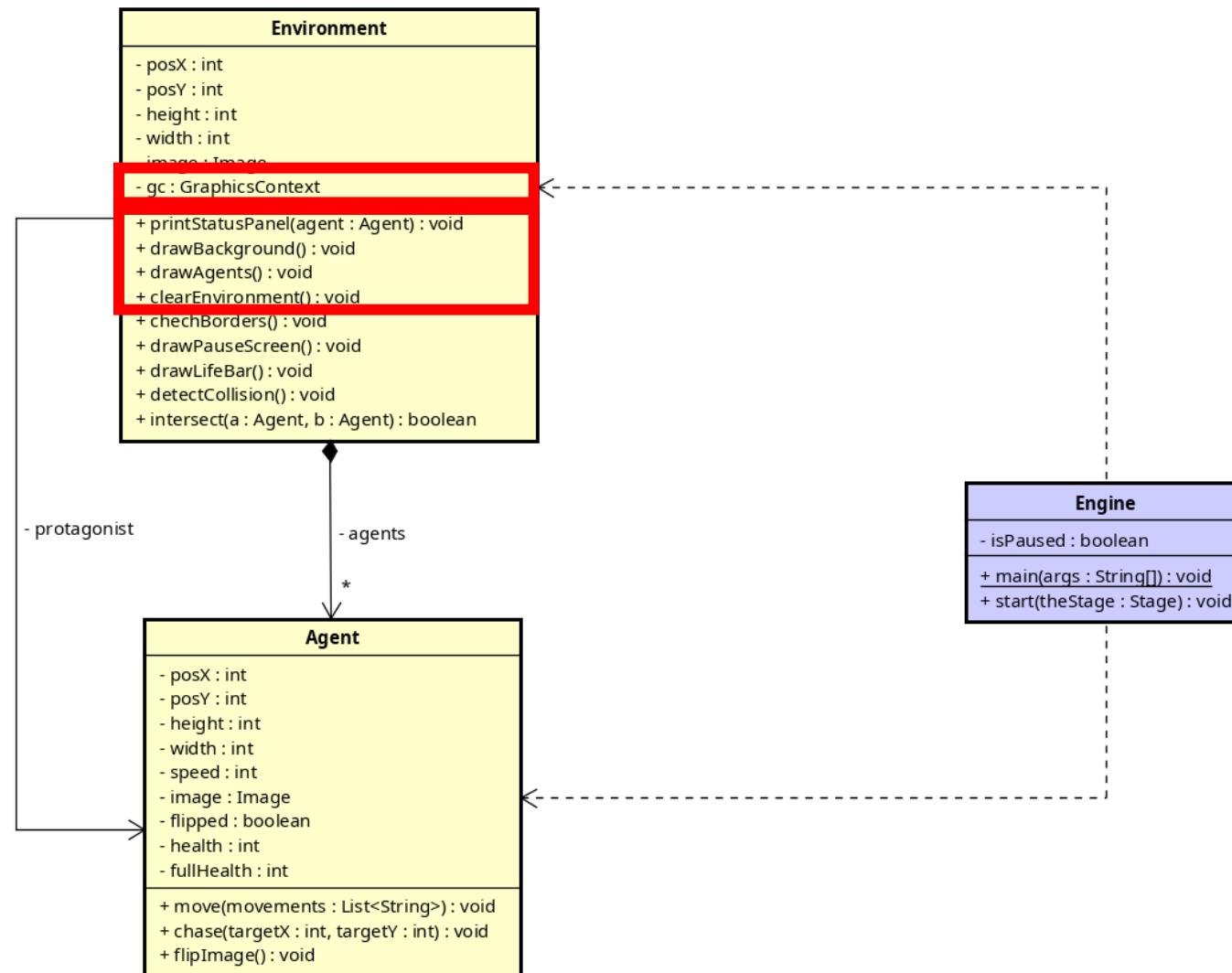
# The Class Diagram



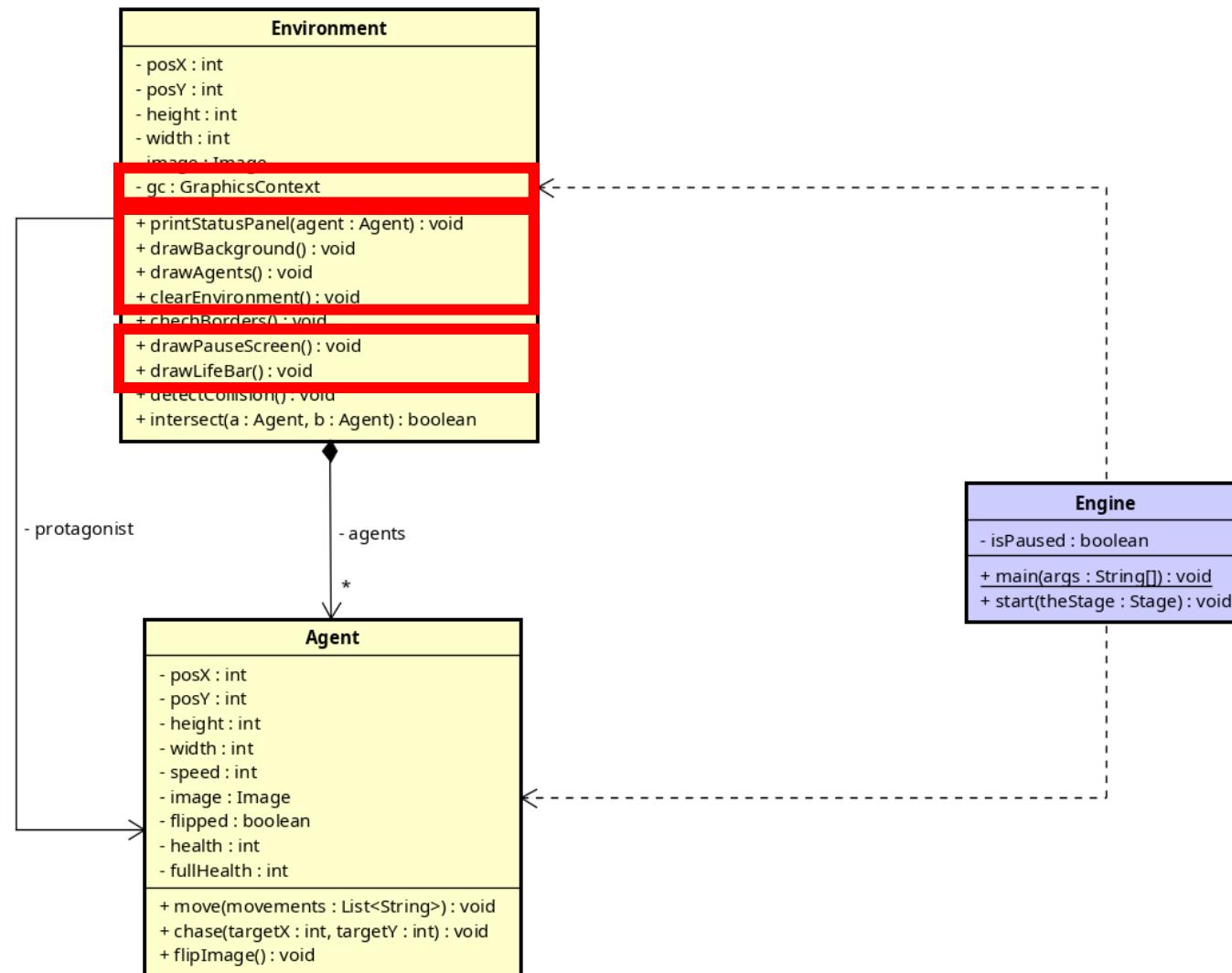
# The Class Diagram



# The Class Diagram

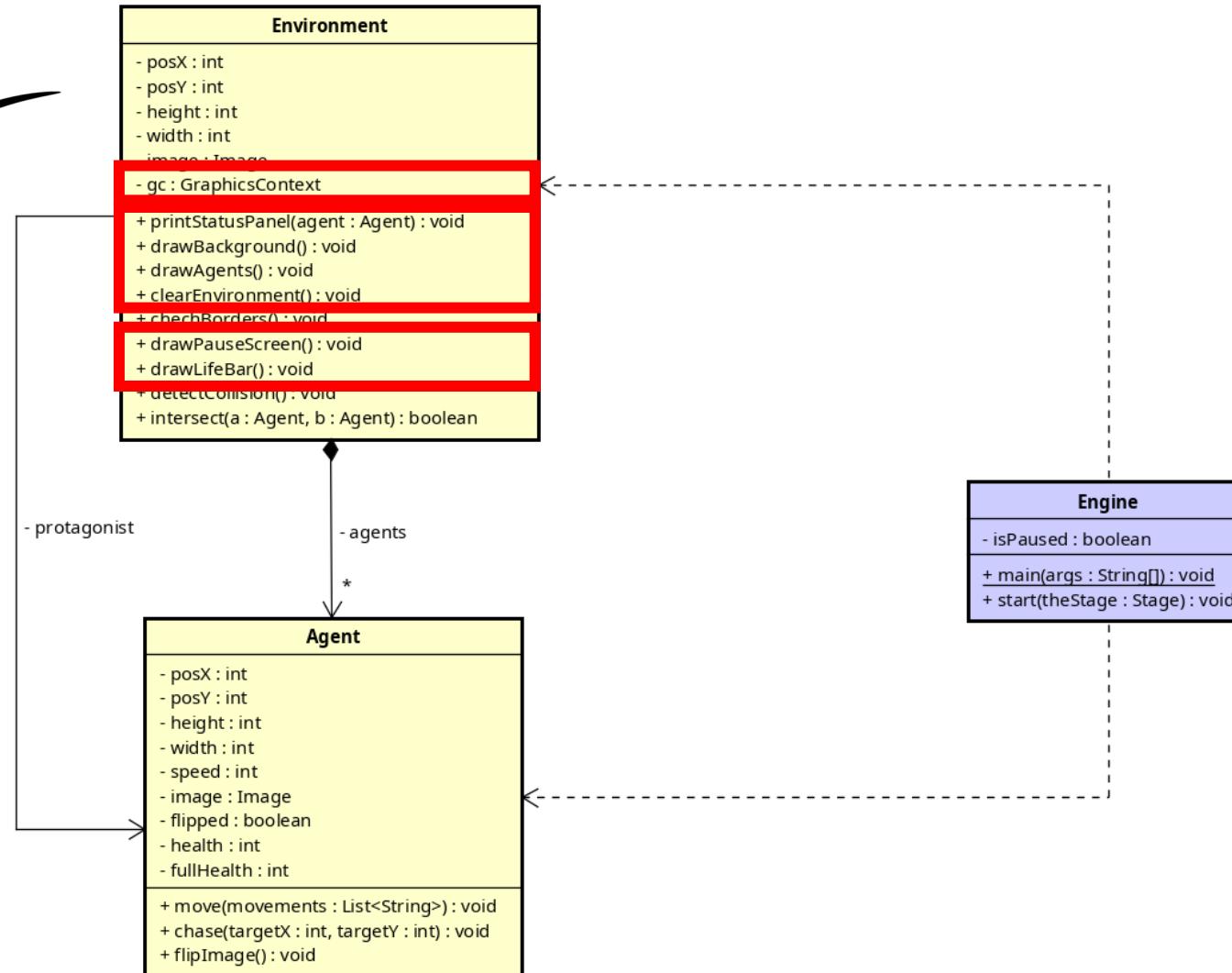


# The Class Diagram



# The Class Diagram

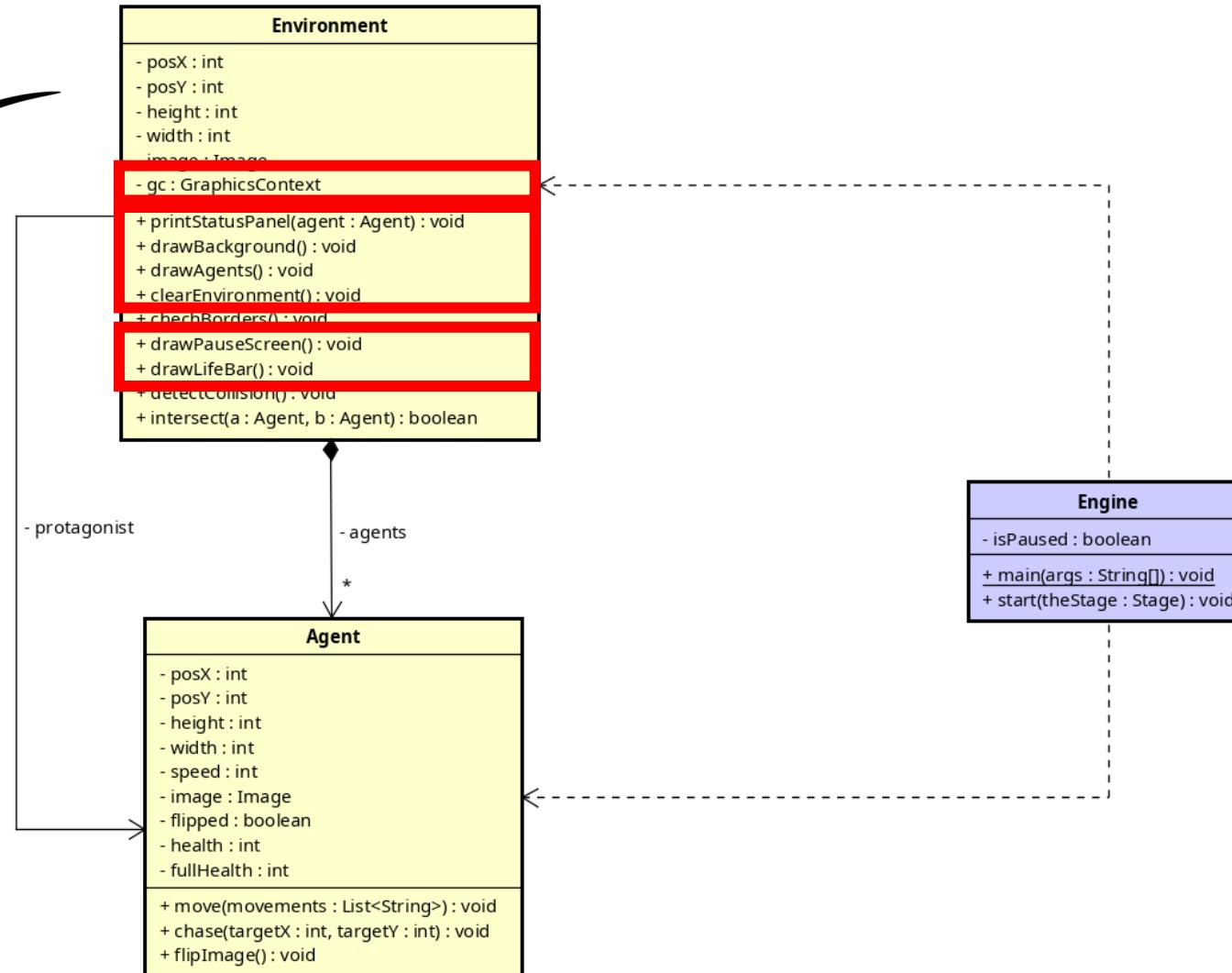
High Coupling  
between Engine  
and JavaFX  
components.



# The Class Diagram

High Coupling  
between Engine  
and JavaFX  
components.

As the Engine  
grows, it gets  
hard to maintain  
and expand.



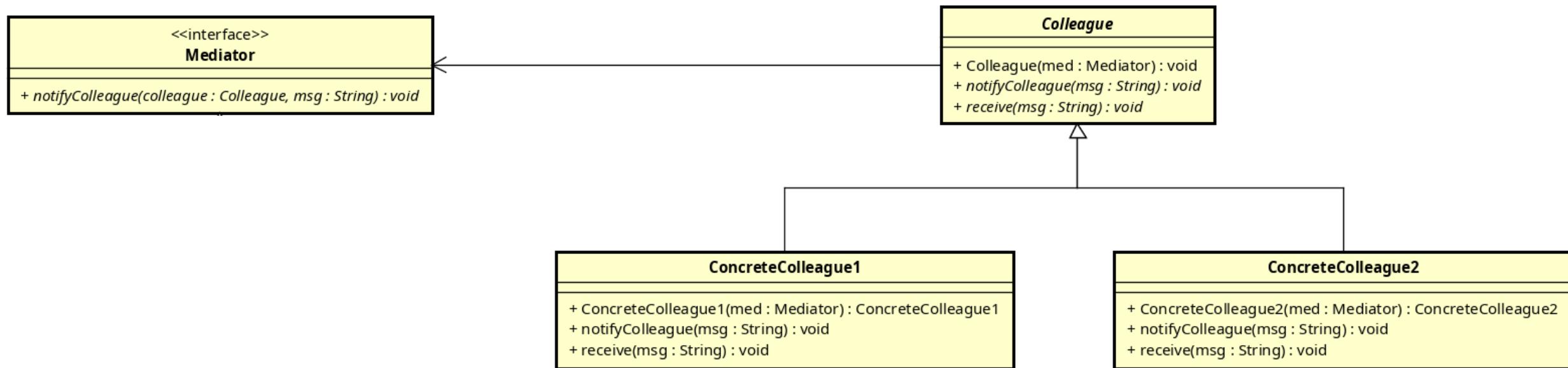
# The Mediator Pattern

The **mediator pattern** is a behavioral design pattern that reduces direct dependencies between components by introducing a mediator object that handles communication between them.

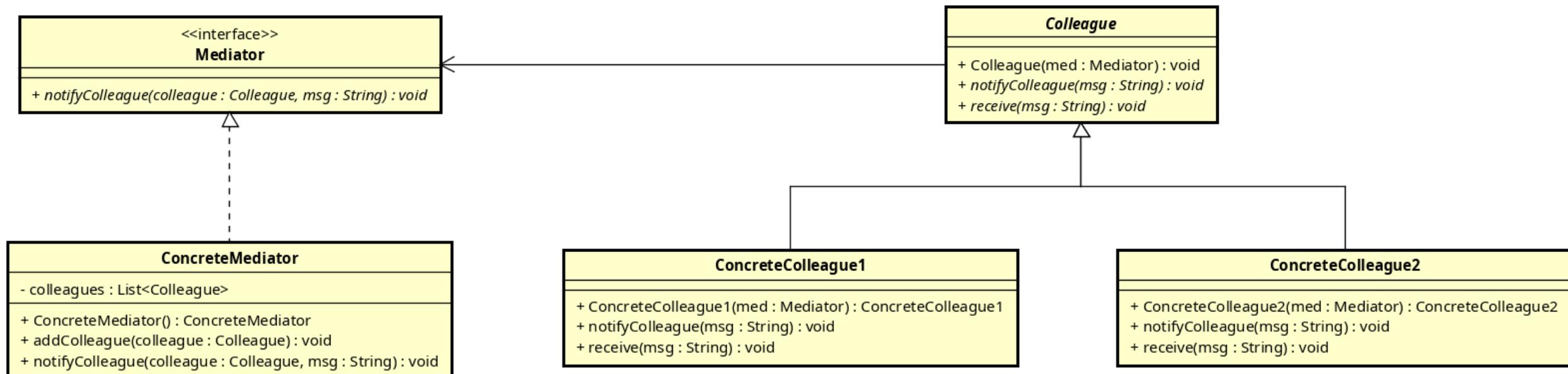
# The Mediator Pattern



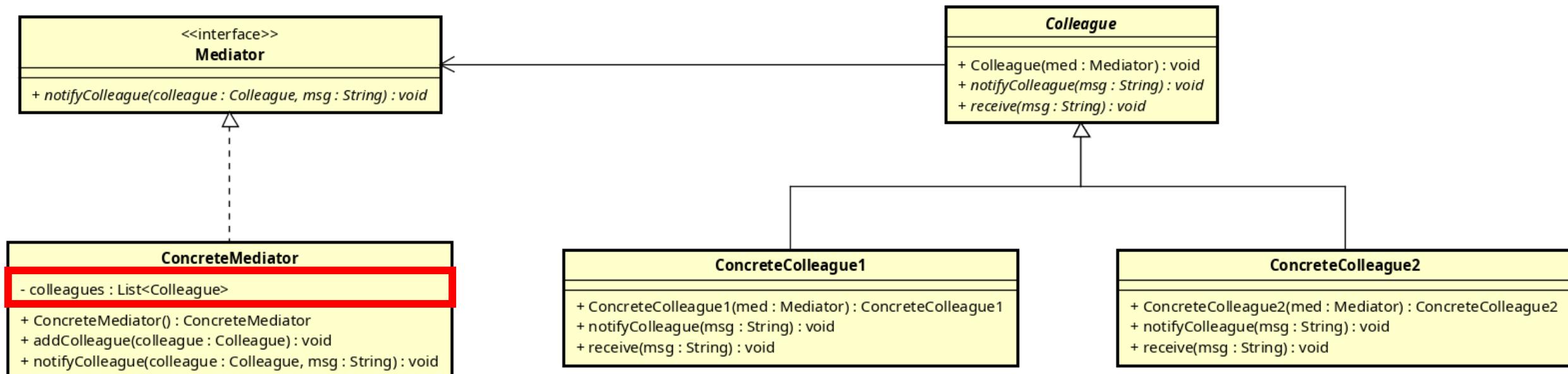
# The Mediator Pattern



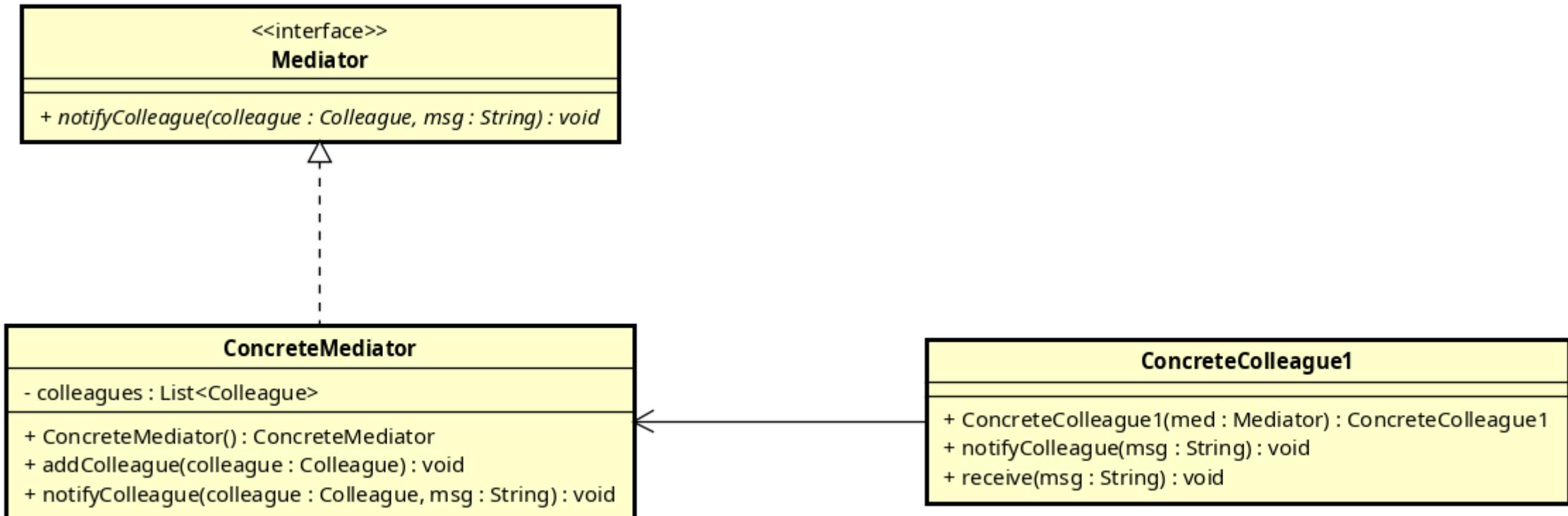
# The Mediator Pattern



# The Mediator Pattern



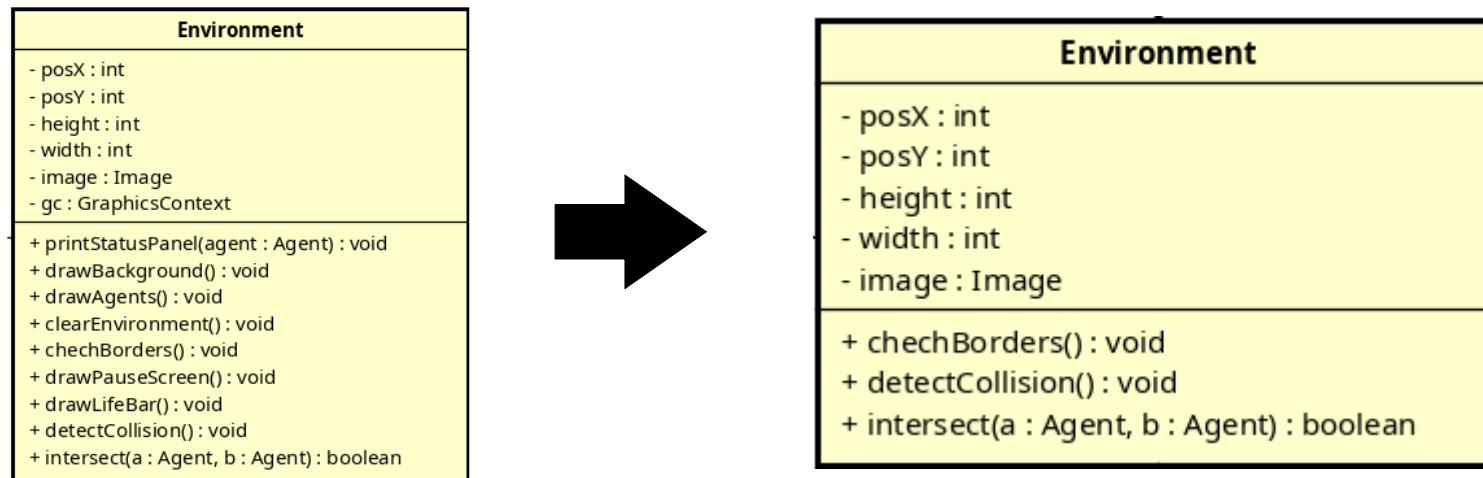
# The Mediator Pattern Simplified



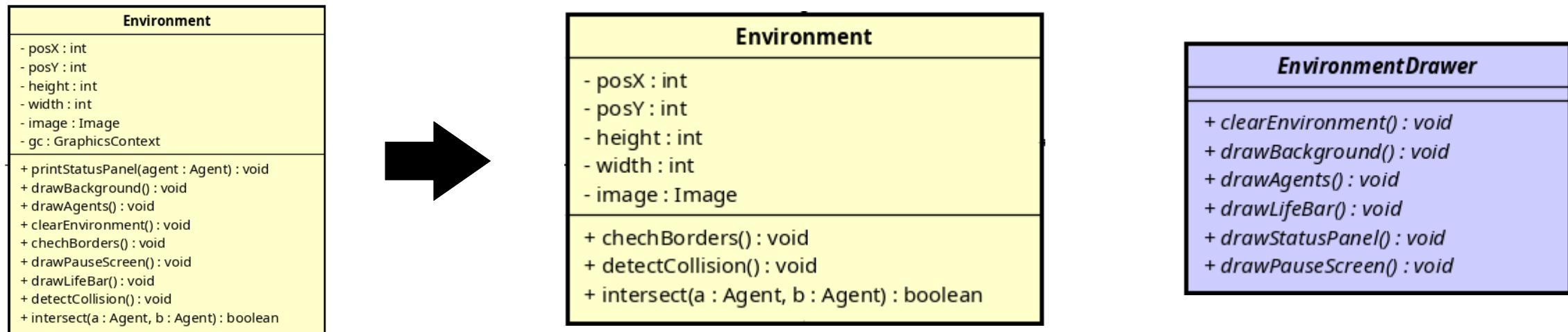
# The Class Diagram

Environment
<pre>- posX : int - posY : int - height : int - width : int - image : Image - gc : GraphicsContext  + printStatusPanel(agent : Agent) : void + drawBackground() : void + drawAgents() : void + clearEnvironment() : void + checkBorders() : void + drawPauseScreen() : void + drawLifeBar() : void + detectCollision() : void + intersect(a : Agent, b : Agent) : boolean</pre>

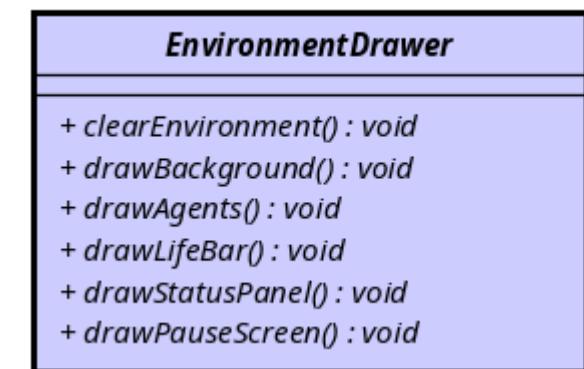
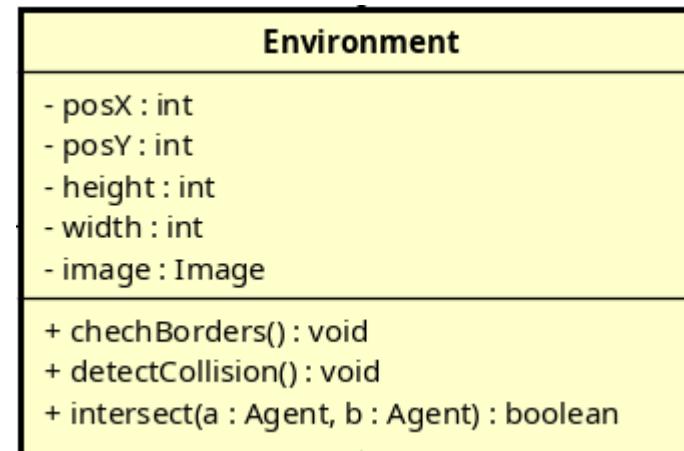
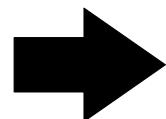
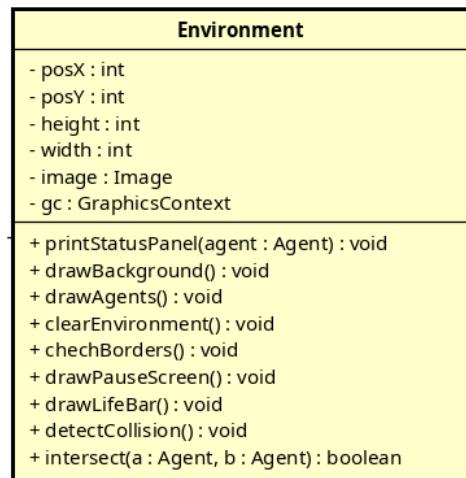
# The Class Diagram



# The Class Diagram

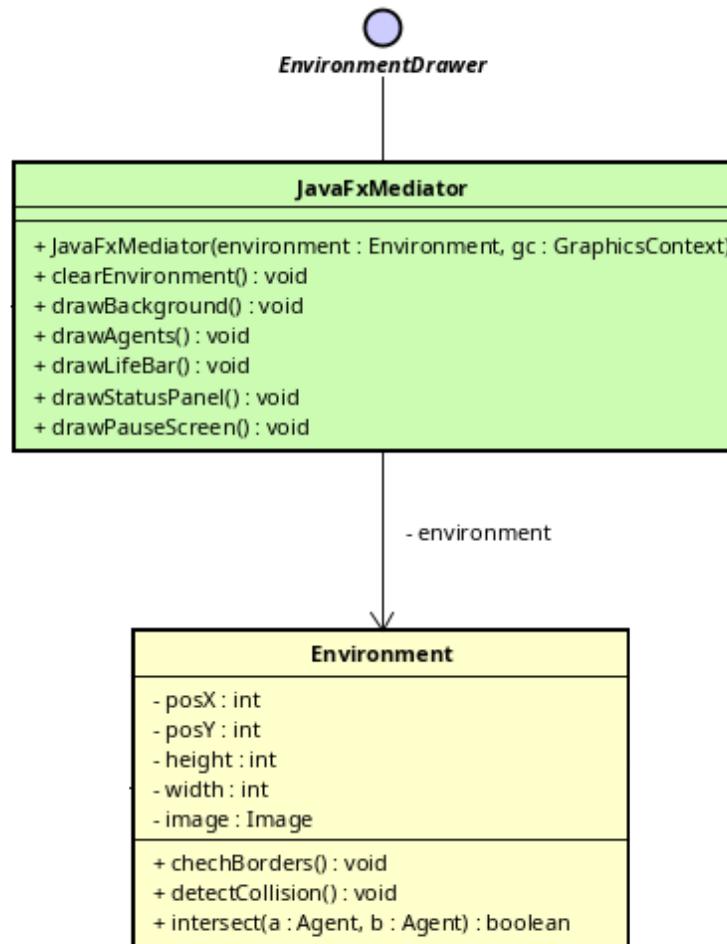


# The Class Diagram

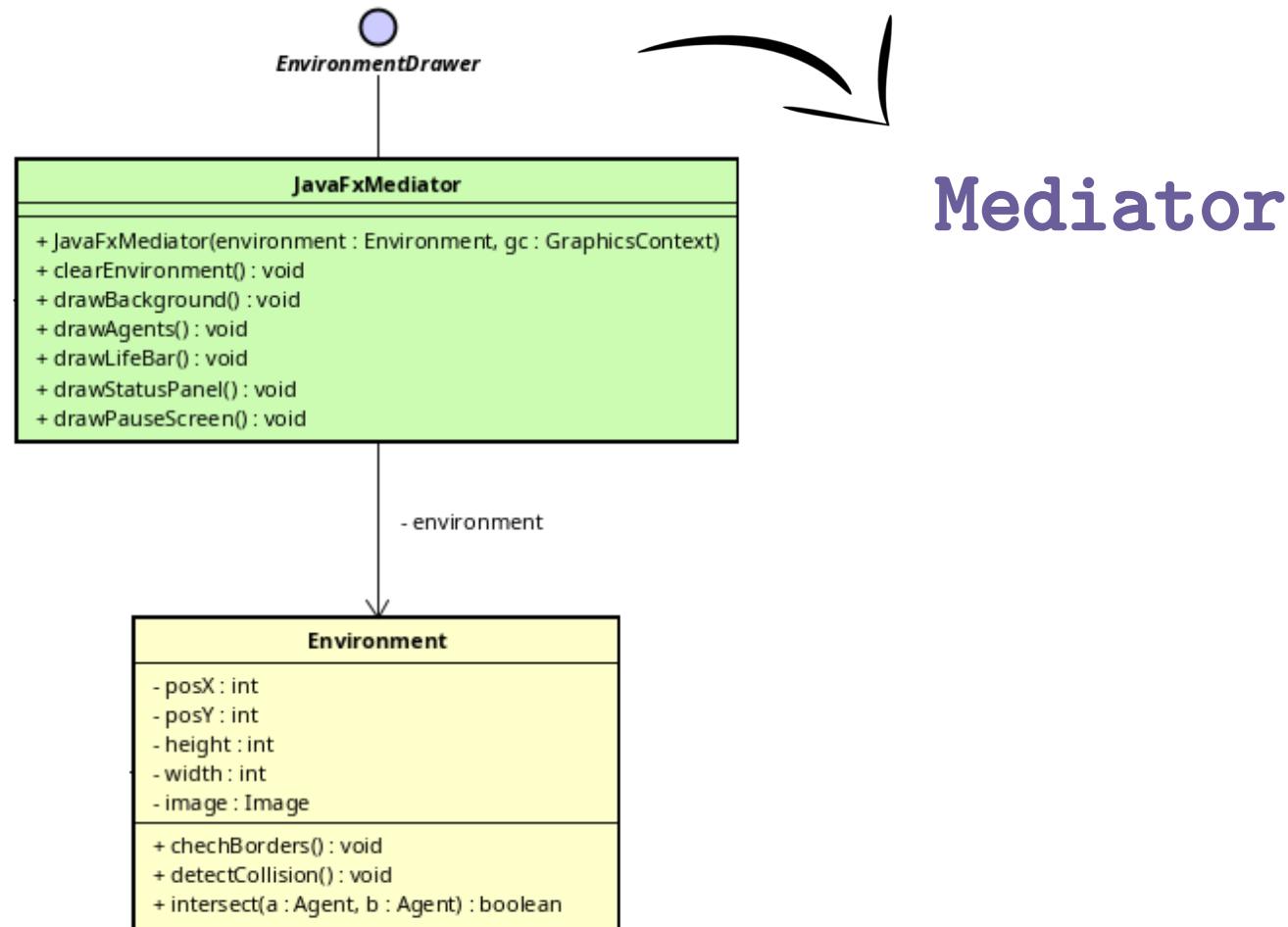


Mediator

# The Class Diagram

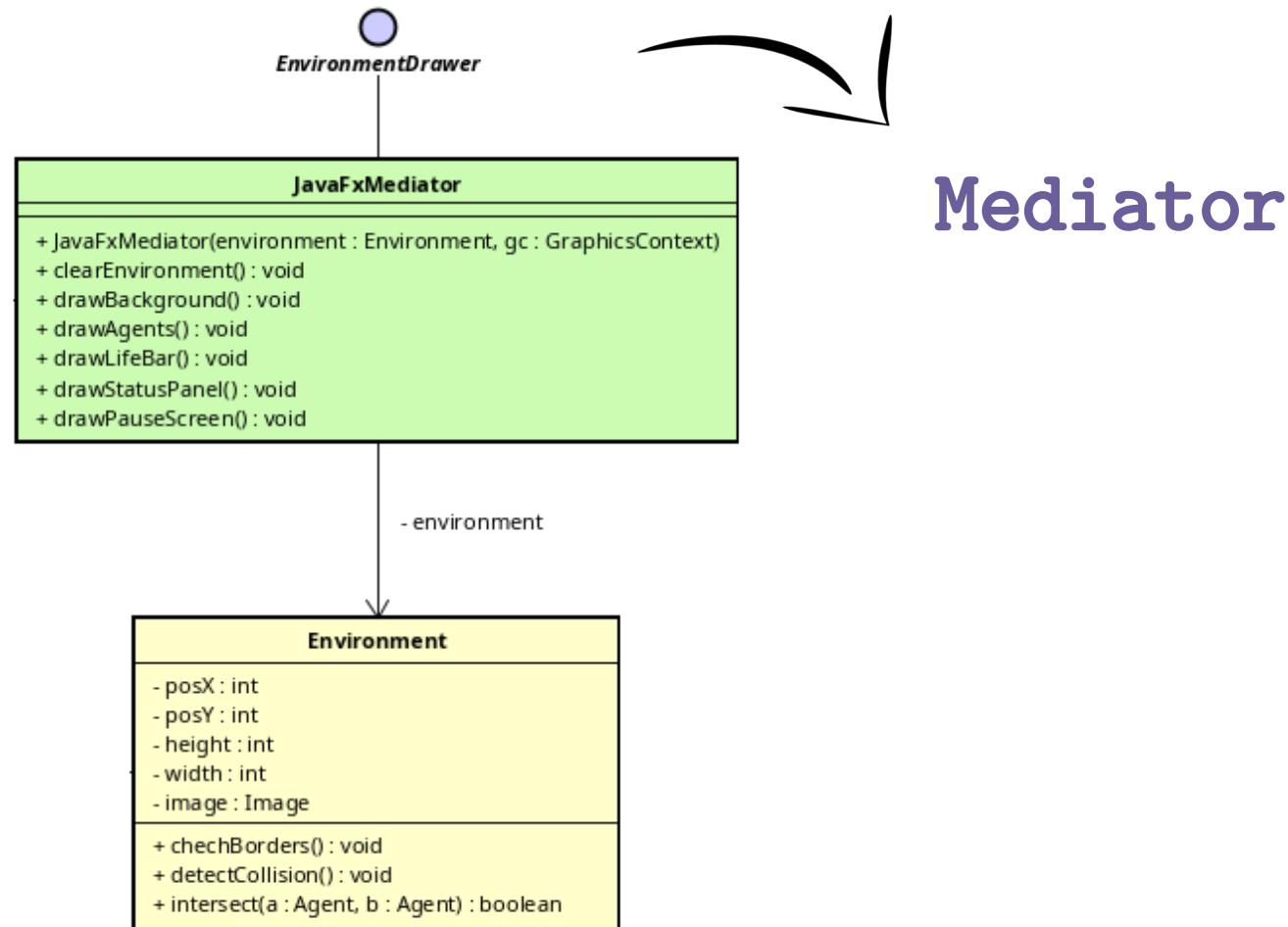


# The Class Diagram



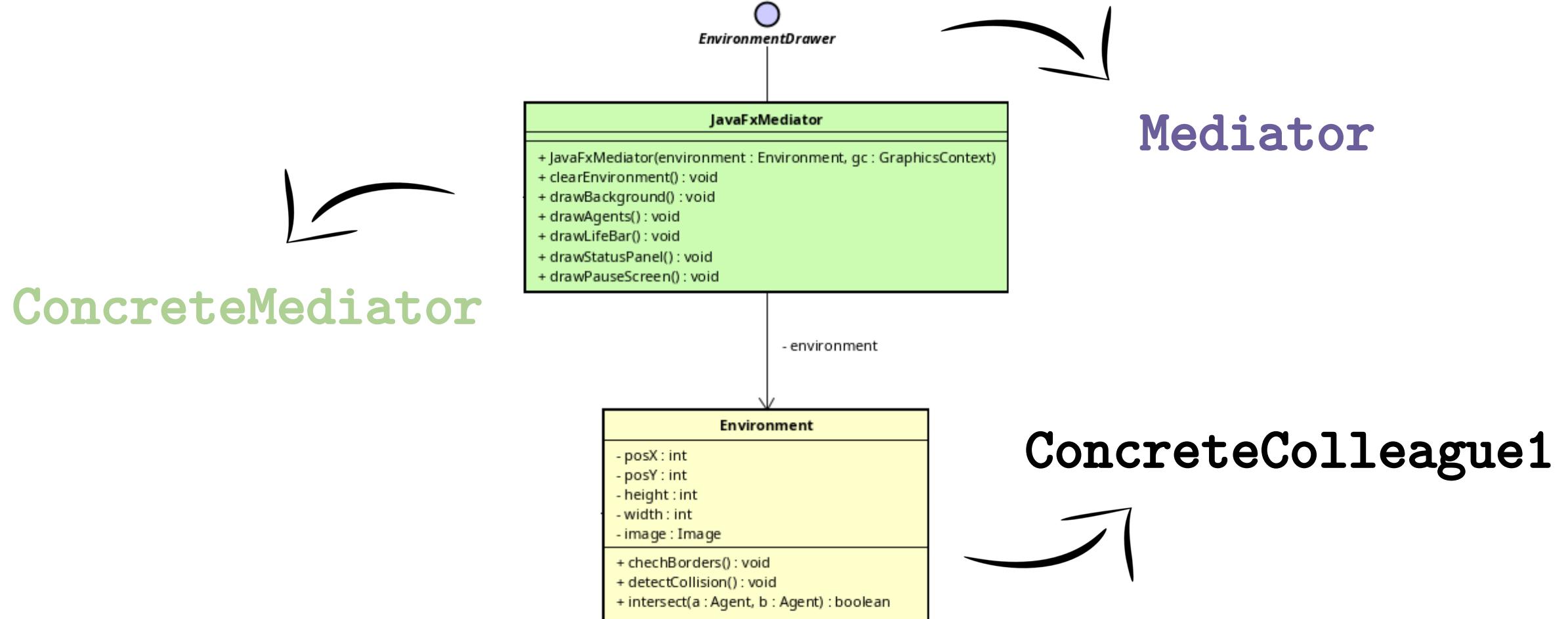
# The Class Diagram

ConcreteMediator

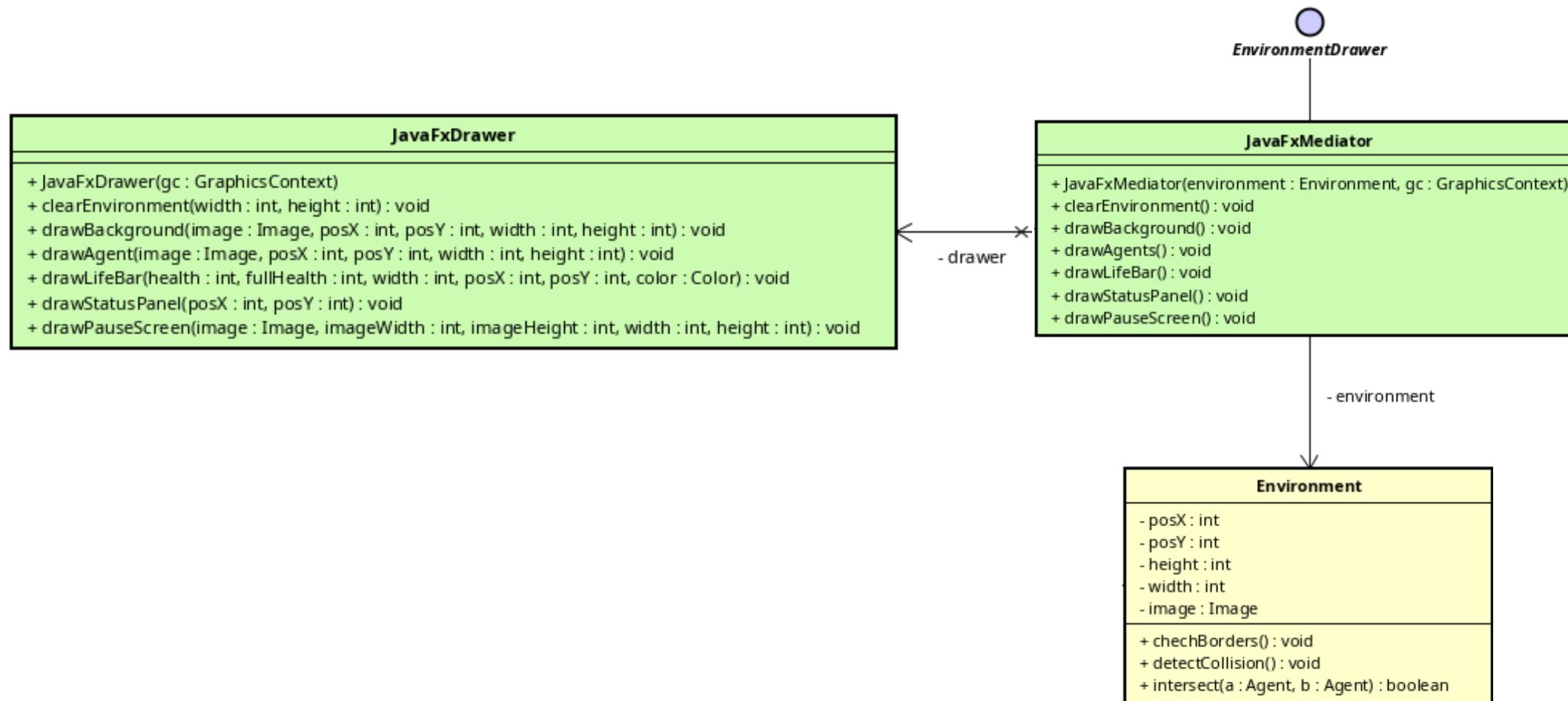


Mediator

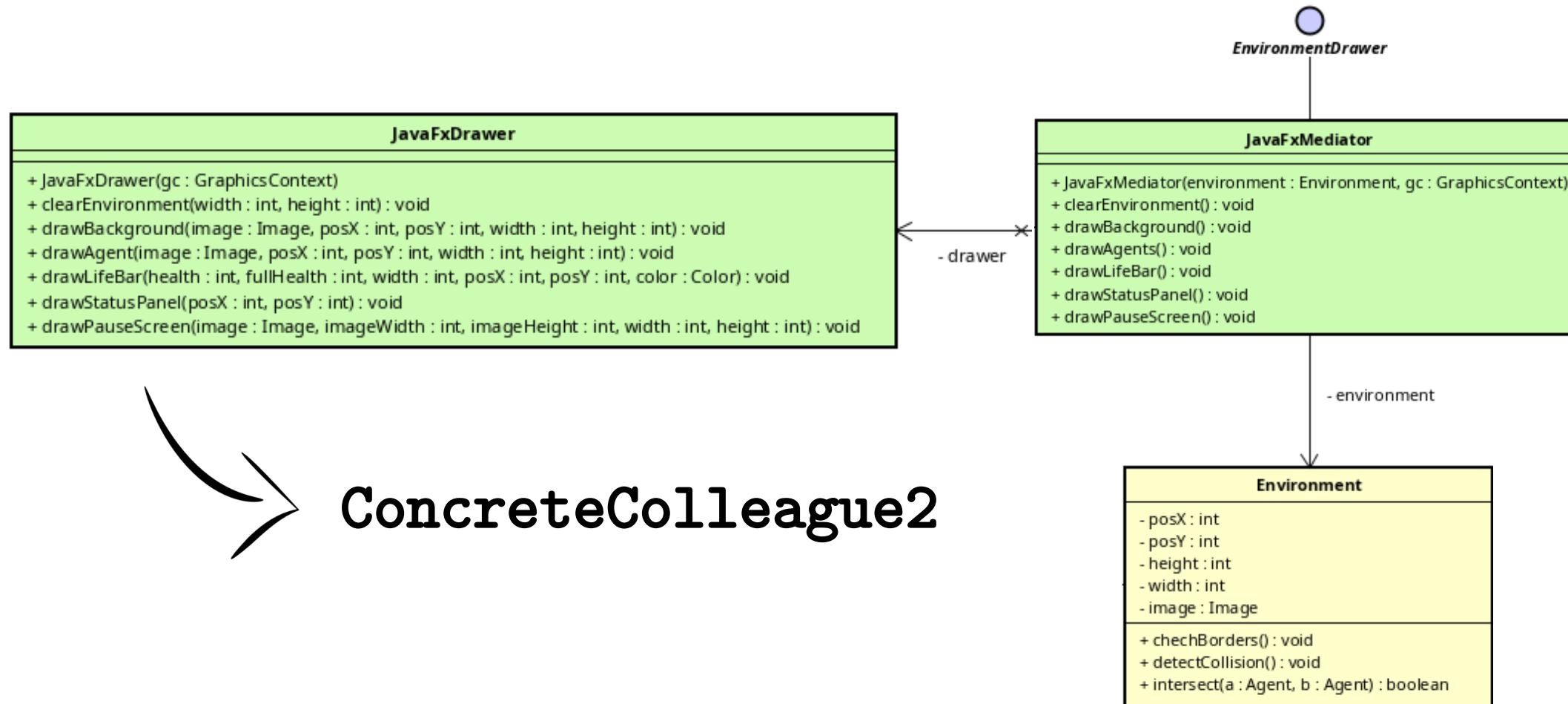
# The Class Diagram



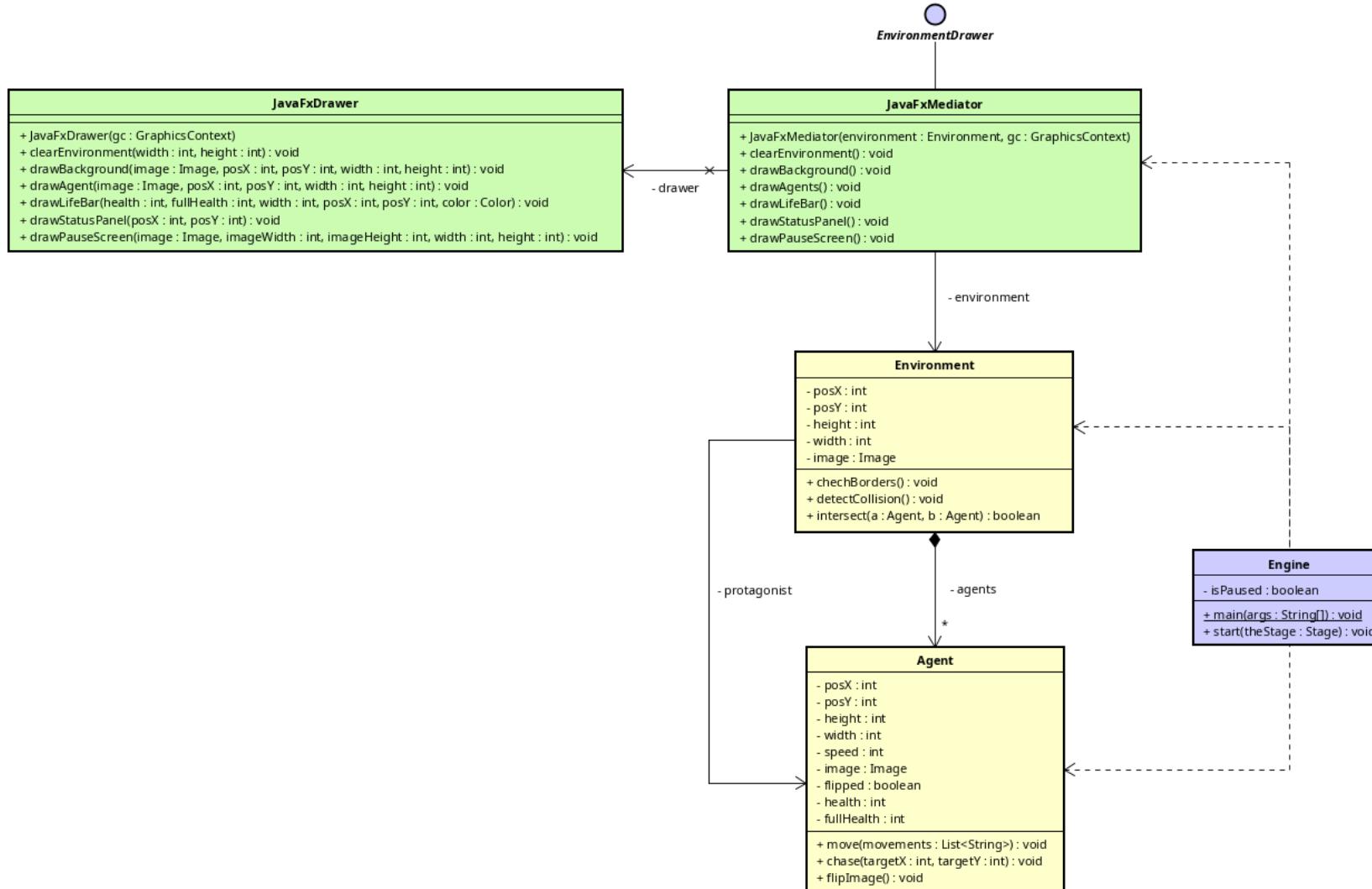
# The Class Diagram



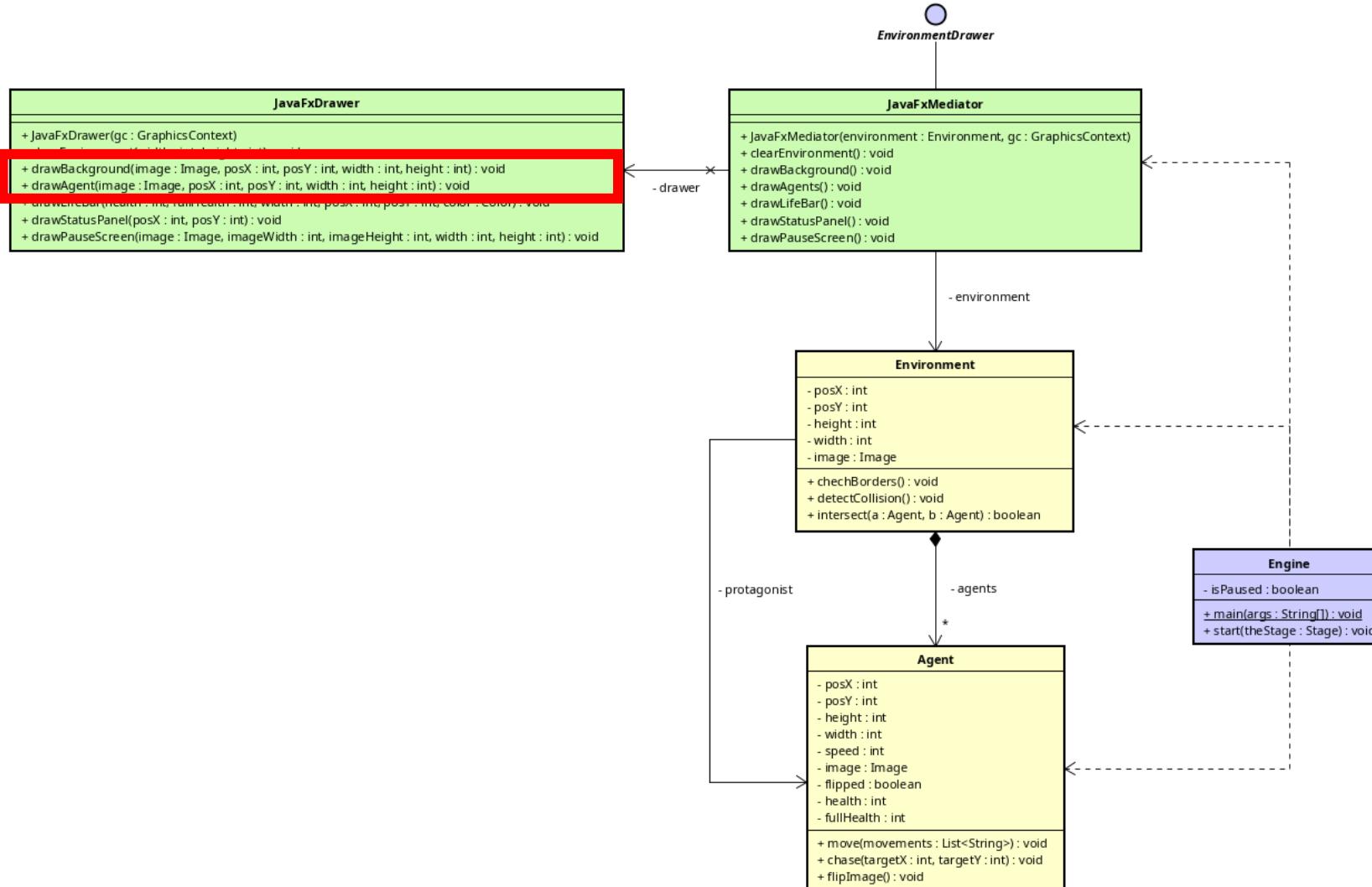
# The Class Diagram



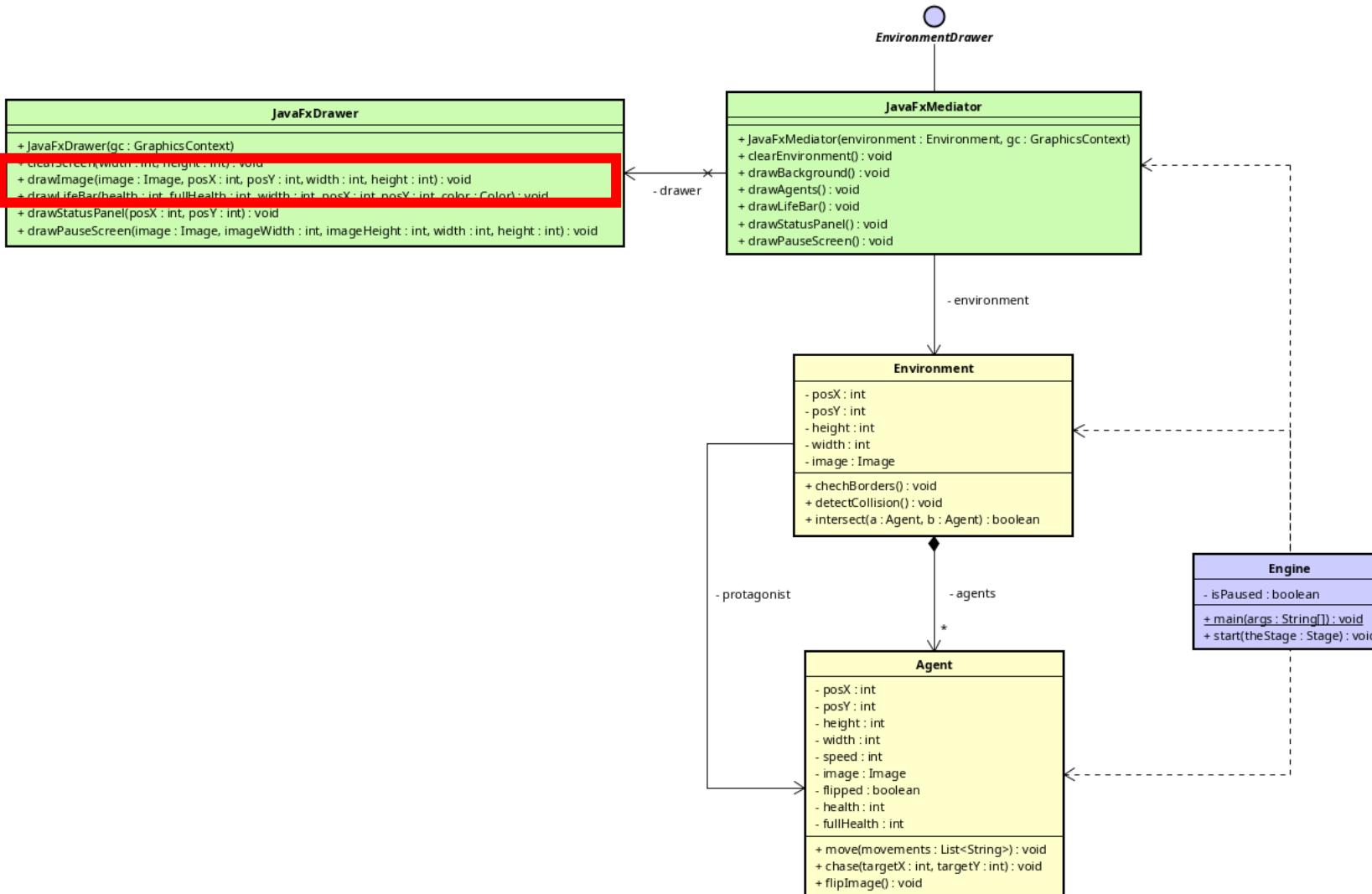
# The Class Diagram



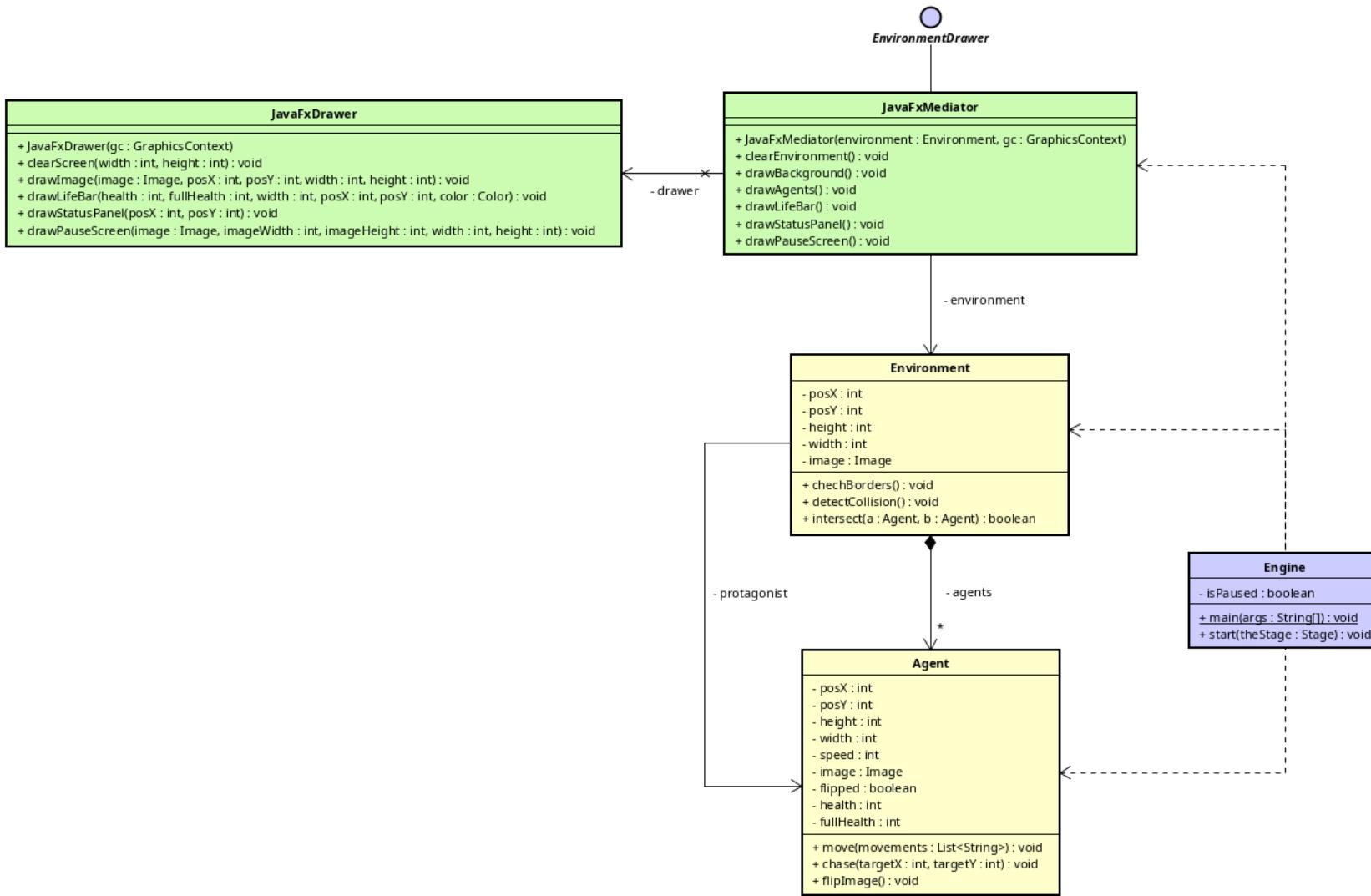
# The Class Diagram



# The Class Diagram: After Refactoring

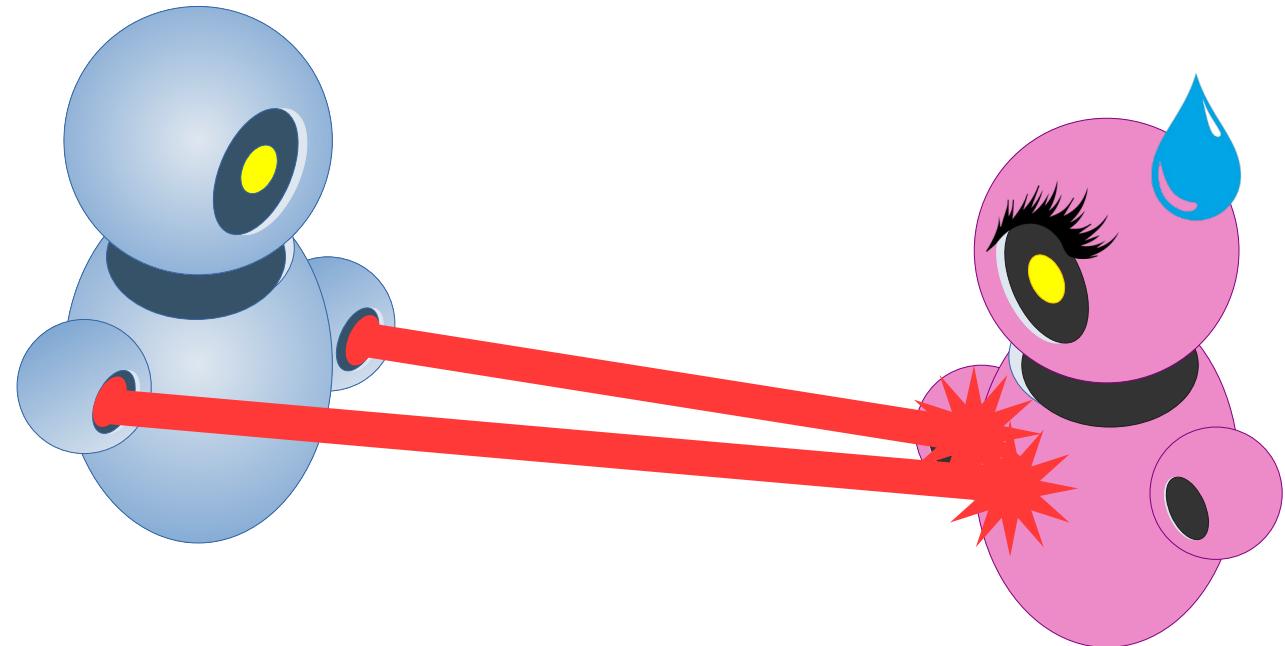


# The Class Diagram: Final Result



# TAKING DAMAGE

*by Collision*

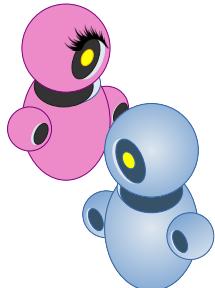


# Types of Damages

An agent can **take damage** by . . .

# Types of Damages

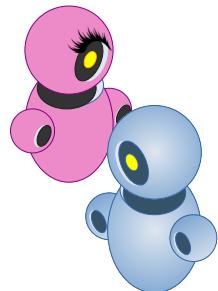
An agent can **take damage** by . . .



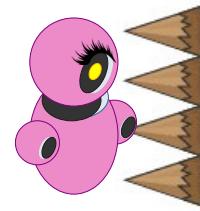
another agent

# Types of Damages

An agent can **take damage** by . . .



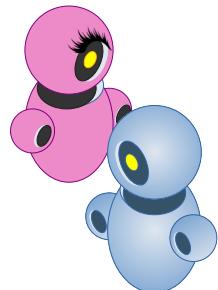
another agent



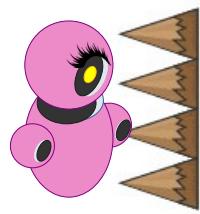
an obstacle

# Types of Damages

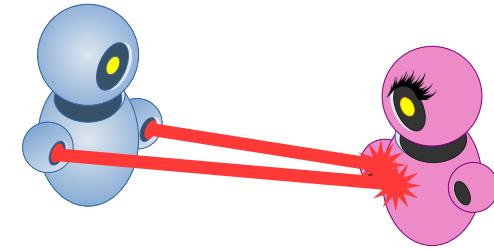
An agent can **take damage** by . . .



another agent



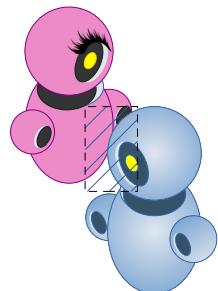
an obstacle



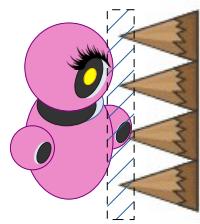
a weapon or power

# Types of Damages

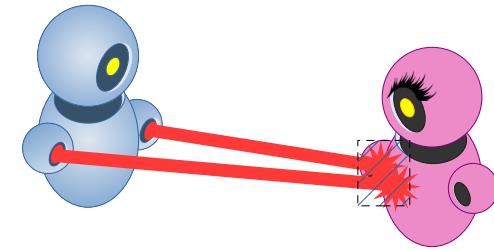
An agent can **take damage** by . . .



another agent



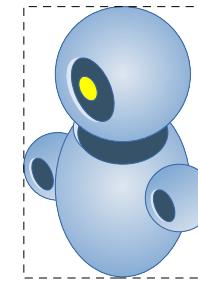
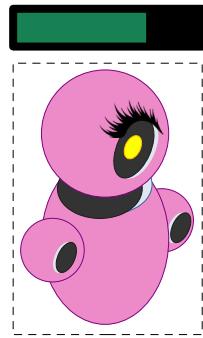
an obstacle



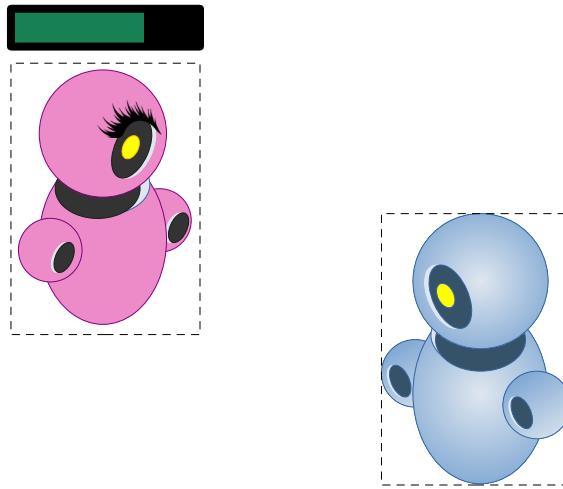
a weapon or power

Using Collision.

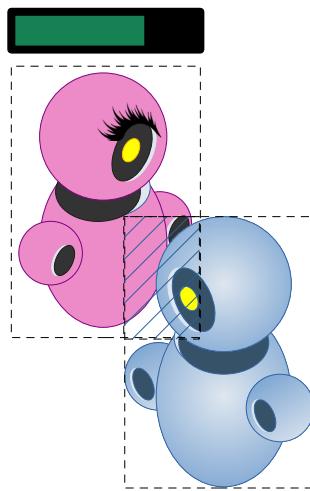
# Collision



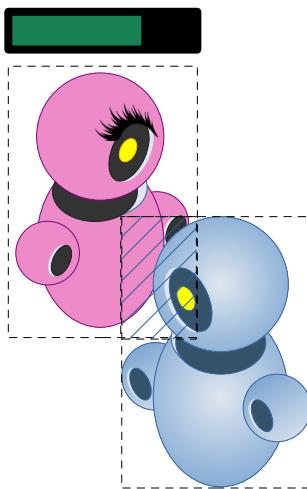
# Collision



# Collision

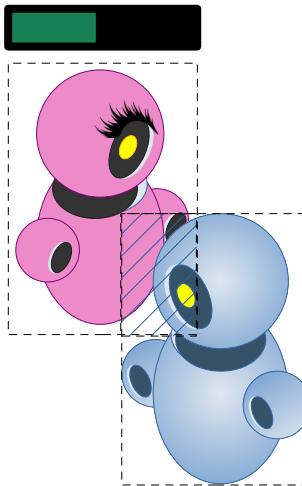


# Collision



collided

# Collision

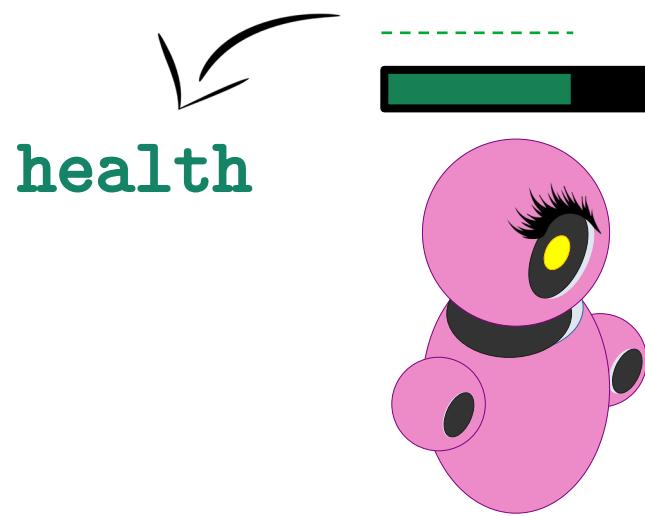


```
if (collided)  
    takeDamage(damage)
```

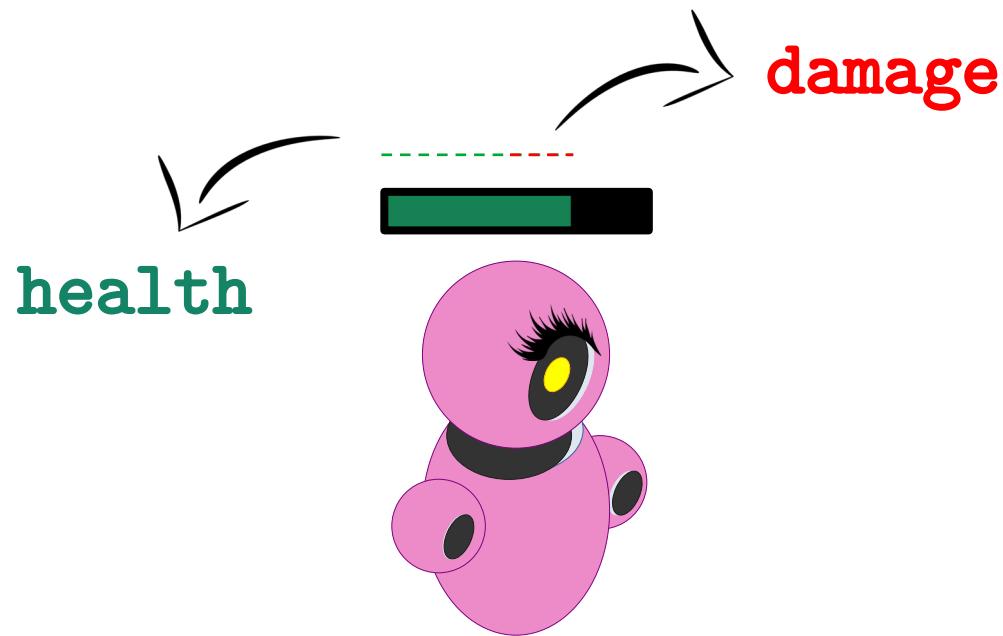
# Taking Damage



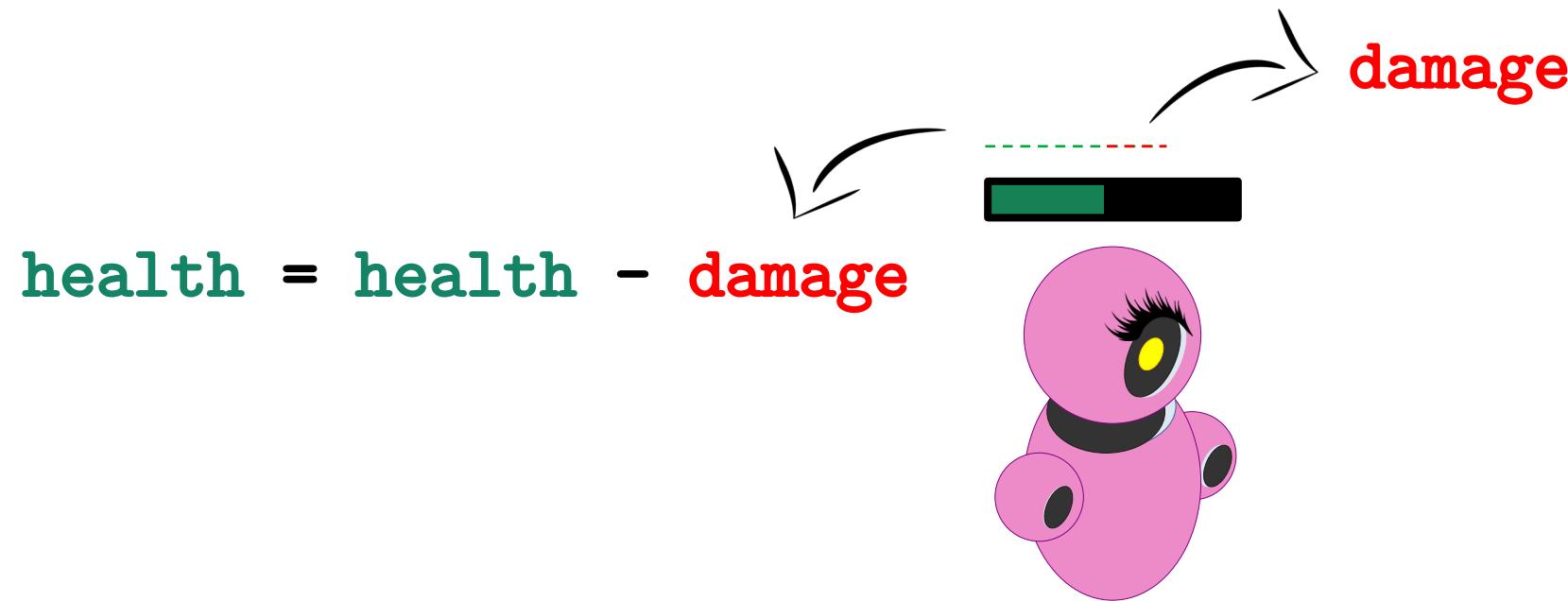
# Taking Damage



# Taking Damage



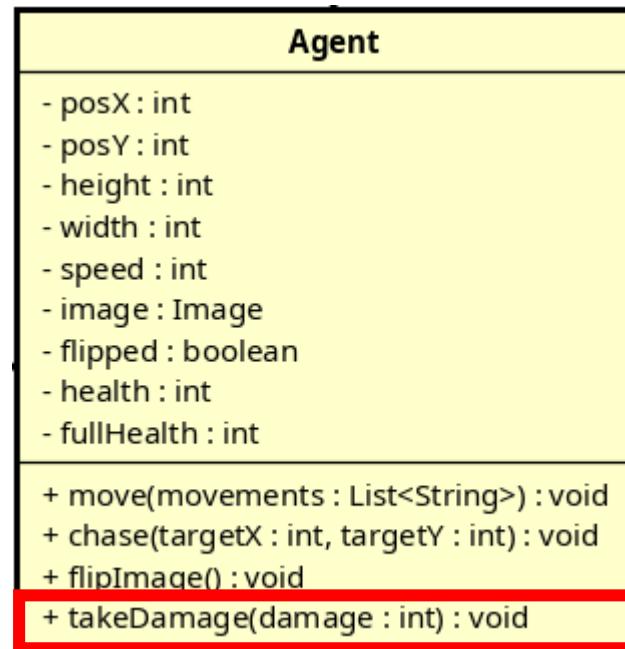
# Taking Damage



# Class Agent: New Method

Agent
<ul style="list-style-type: none"><li>- posX : int</li><li>- posY : int</li><li>- height : int</li><li>- width : int</li><li>- speed : int</li><li>- image : Image</li><li>- flipped : boolean</li><li>- health : int</li><li>- fullHealth : int</li></ul>
<ul style="list-style-type: none"><li>+ move(movements : List&lt;String&gt;) : void</li><li>+ chase(targetX : int, targetY : int) : void</li><li>+ flipImage() : void</li><li>+ takeDamage(damage : int) : void</li></ul>

# Class Agent: New Method



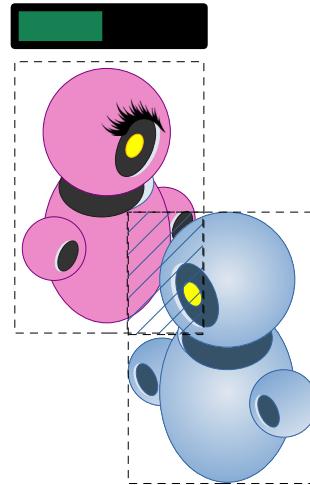
# INVULNERABILITY



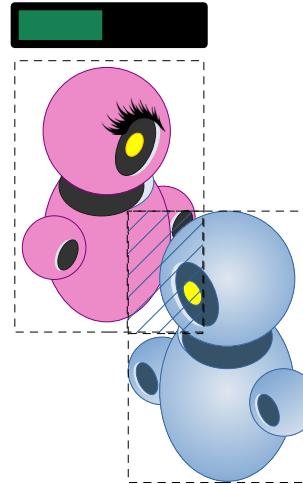
# Invulnerability

Invulnerability is a  
**temporary state** where damages  
do not apply.

# Invulnerability

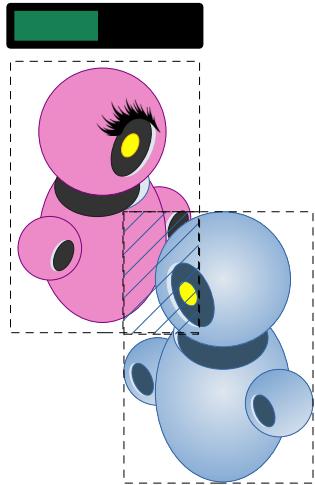


# Invulnerability



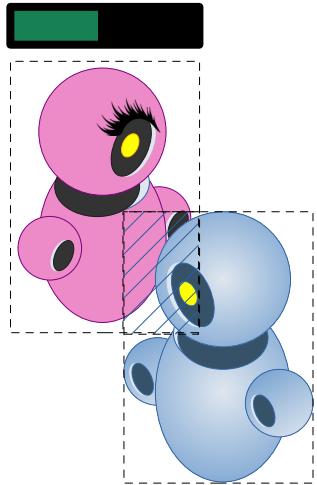
```
if (collided)  
    updateInvulnerability()  
    takeDamage(damage)
```

# Invulnerability



To update the **invulnerability** status:

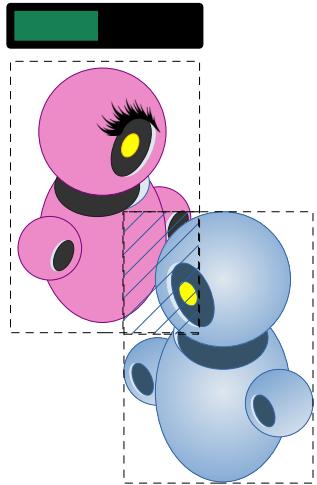
# Invulnerability



To update the **invulnerability** status:

1. the last hit time;

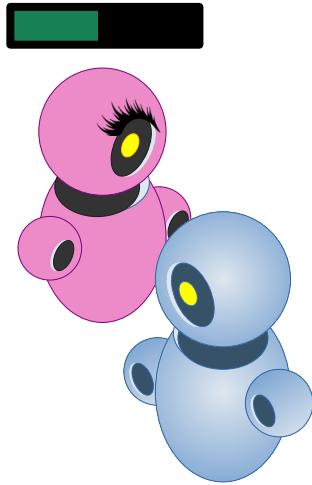
# Invulnerability



To update the **invulnerability** status:

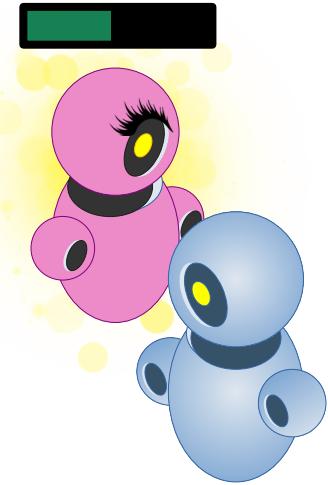
1. the last hit time;
2. the cooldown time.

# Invulnerability



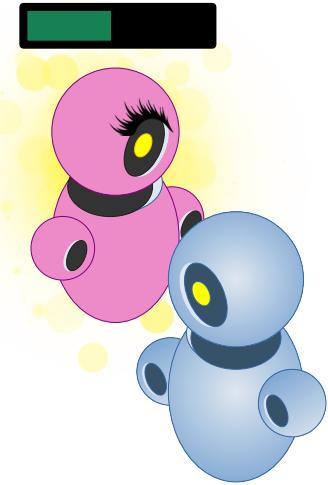
`(now - lastHitTime) < cooldownTime)`

# Invulnerability



```
(now - lastHitTime) < cooldownTime)  
invulnerable = true
```

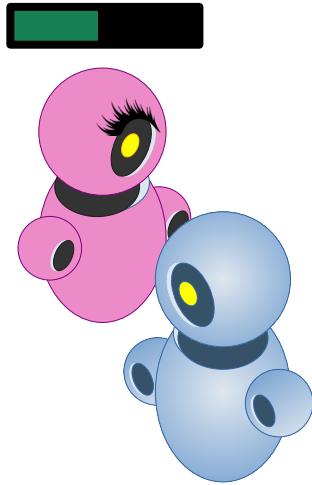
# Invulnerability



```
(now - lastHitTime) < cooldownTime)  
invulnerable = true
```

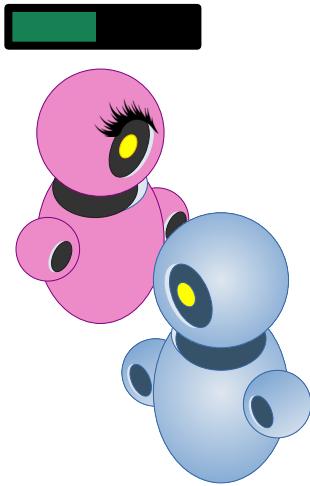
Nothing happens !

# Invulnerability



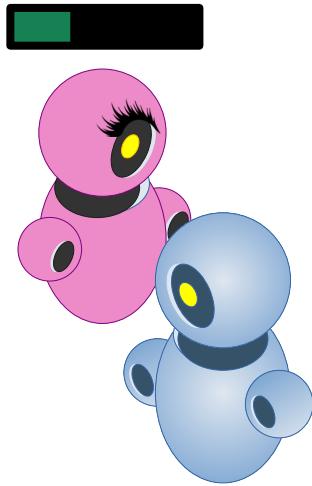
`(now - lastHitTime) > cooldownTime)`

# Invulnerability



```
(now - lastHitTime) > cooldownTime)  
Invulnerable = false
```

# Invulnerability



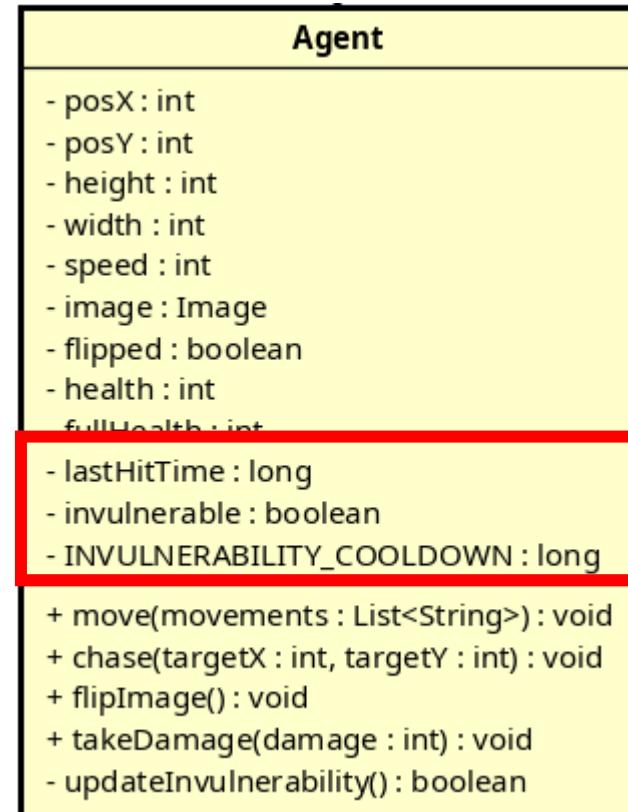
```
(now - lastHitTime) > cooldownTime)  
Invulnerable = false
```

It gets damage!

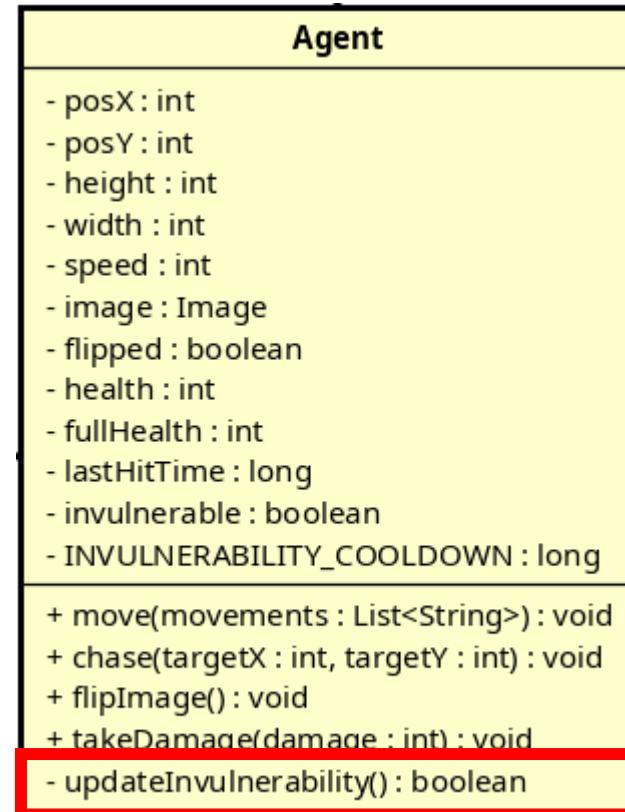
# Class Agent: New Attributes and Method

Agent	
- posX : int	
- posY : int	
- height : int	
- width : int	
- speed : int	
- image : Image	
- flipped : boolean	
- health : int	
- fullHealth : int	
- lastHitTime : long	
- invulnerable : boolean	
- INVULNERABILITY_COOLDOWN : long	
+ move(movements : List<String>) : void	
+ chase(targetX : int, targetY : int) : void	
+ flipImage() : void	
+ takeDamage(damage : int) : void	
- updateInvulnerability() : boolean	

# Class Agent: New Attributes and Method



# Class Agent: New Attributes and Method



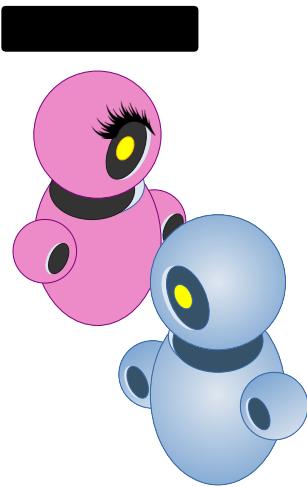
**THE GAME IS  
OVER**



# The Game Over

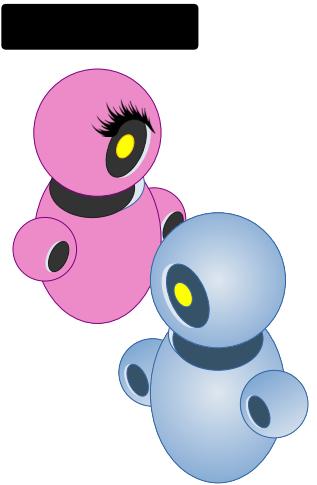
The game is **over** when  
the protagonist dies.

# The Game Over



```
if(this.getHealth() <= 0)
```

# The Game Over



```
if(this.getHealth() <= 0)
```

The agent is dead!

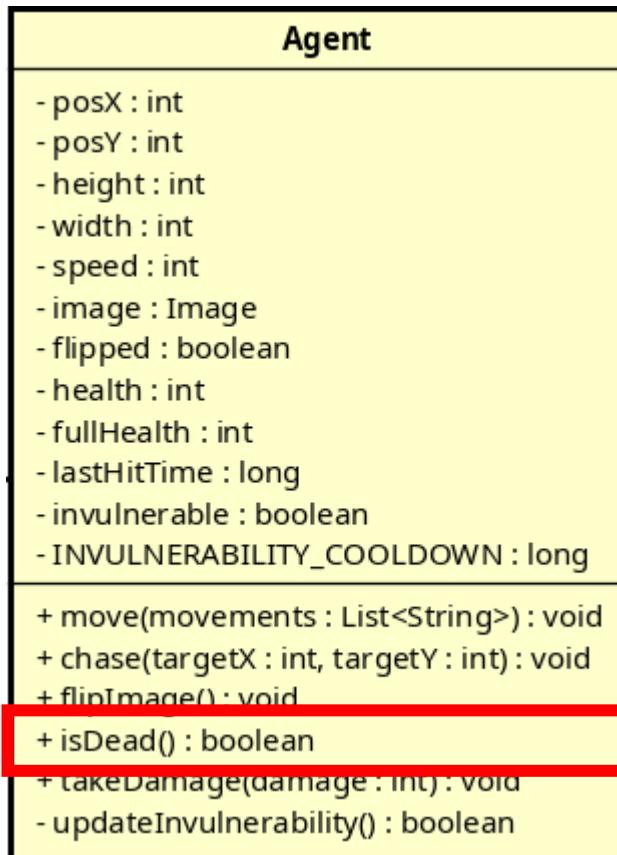
# In The Game Loop

```
if (protagonist is dead)
    draw game over
else
    if (isPaused)
        game paused
        draw pause screen
    else
        game dynamics
```

# Class Agent: New Method

Agent	
- posX : int	
- posY : int	
- height : int	
- width : int	
- speed : int	
- image : Image	
- flipped : boolean	
- health : int	
- fullHealth : int	
- lastHitTime : long	
- invulnerable : boolean	
- INVULNERABILITY_COOLDOWN : long	
+ move(movements : List<String>) : void	
+ chase(targetX : int, targetY : int) : void	
+ flipImage() : void	
+ isDead() : boolean	
+ takeDamage(damage : int) : void	
- updateInvulnerability() : boolean	

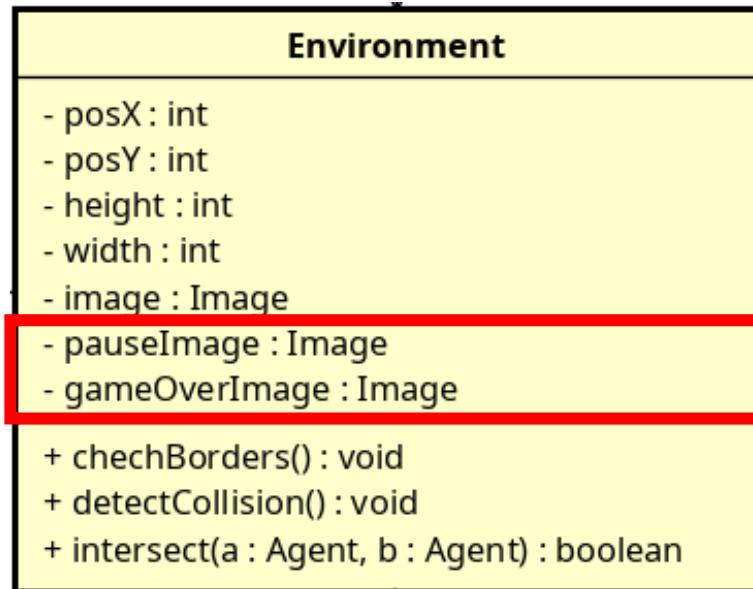
# Class Agent: New Method



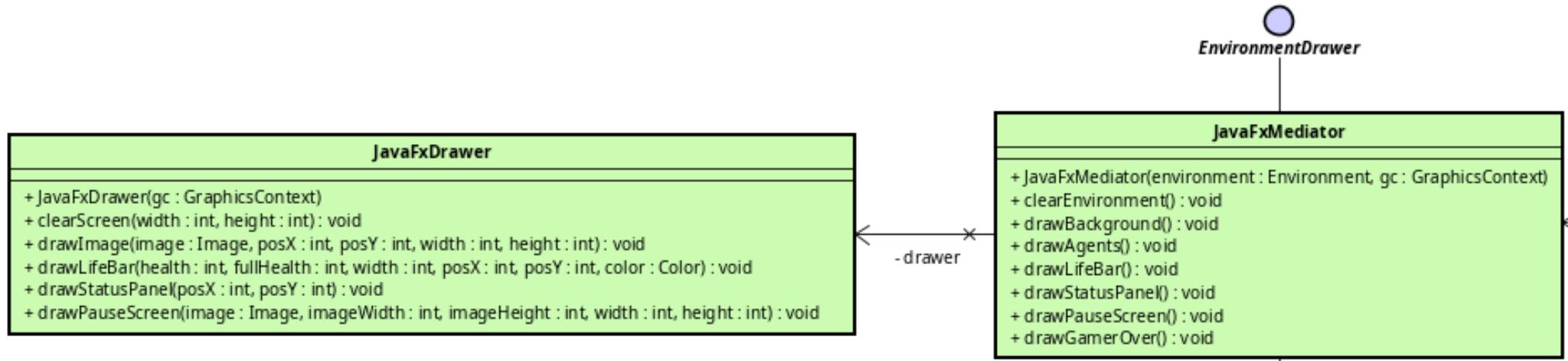
# Class Environment: New Attributes

Environment
<ul style="list-style-type: none"><li>- posX : int</li><li>- posY : int</li><li>- height : int</li><li>- width : int</li><li>- image : Image</li><li>- pauseImage : Image</li><li>- gameOverImage : Image</li></ul>
<ul style="list-style-type: none"><li>+ checkBorders() : void</li><li>+ detectCollision() : void</li><li>+ intersect(a : Agent, b : Agent) : boolean</li></ul>

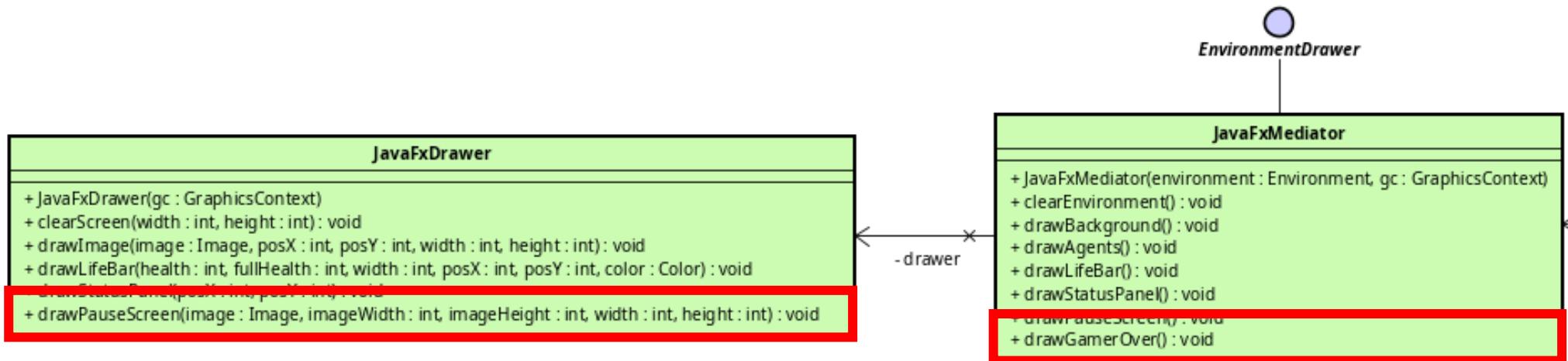
# Class Environment: New Attributes



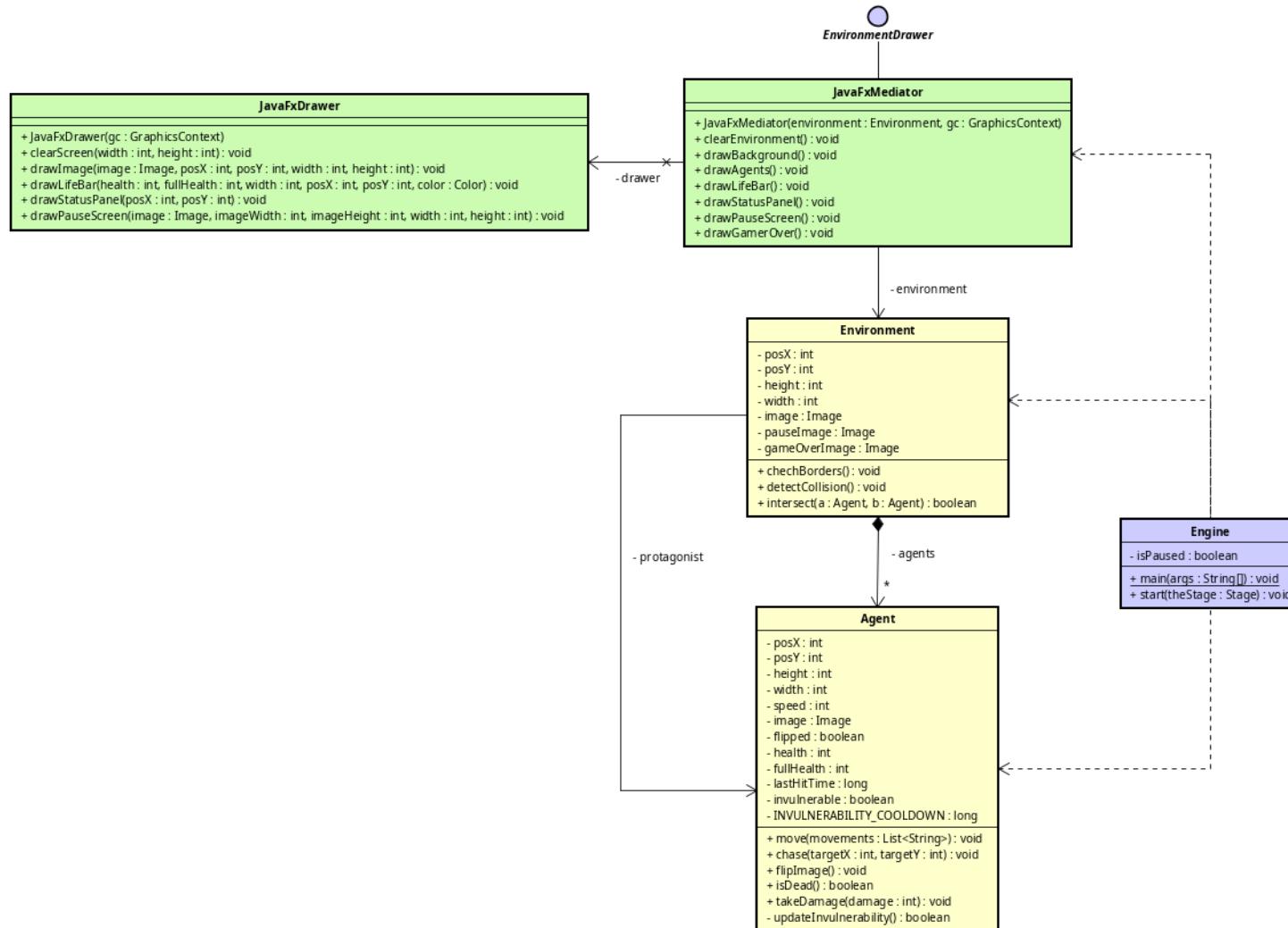
# The Mediator and Drawer Classes: New Method



# The Mediator and Drawer Classes: New Method



# The Modified Model



# THE GENERAL ENTITY



# The Entity

The entity is **any printable** instance  
that must appear in the screen.

# The Entity



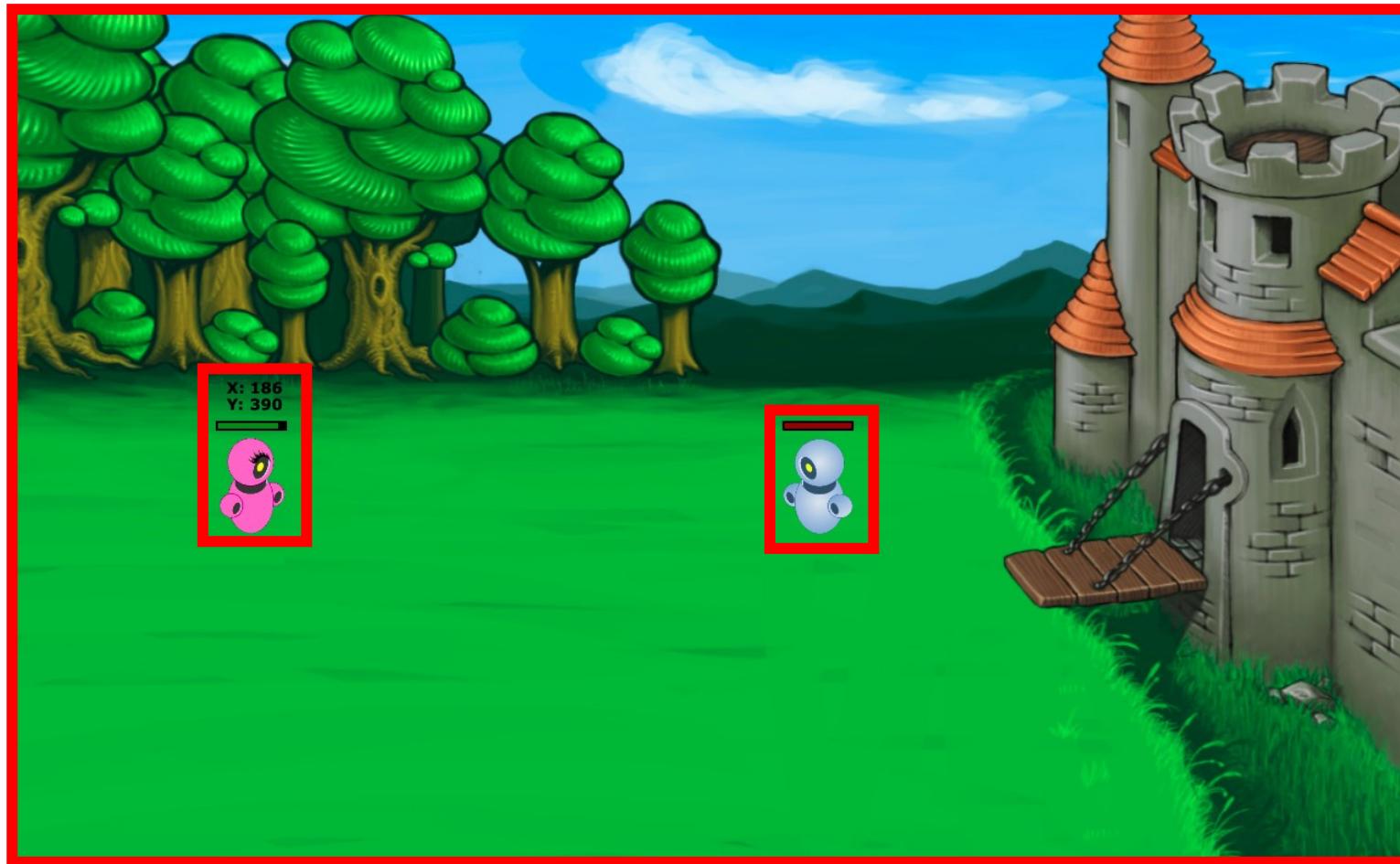
# The Entity



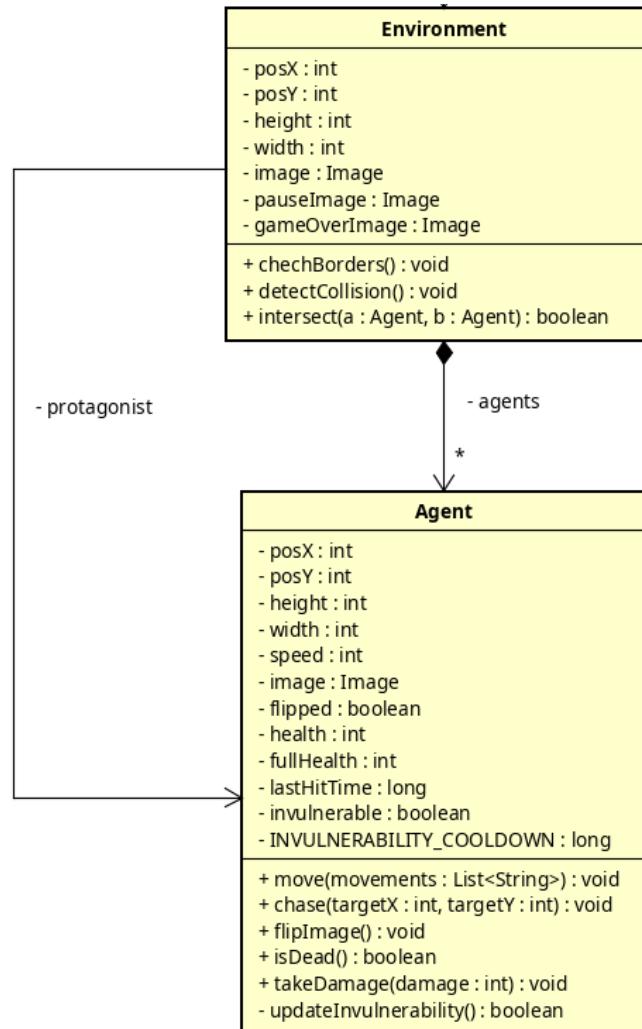
# The Entity



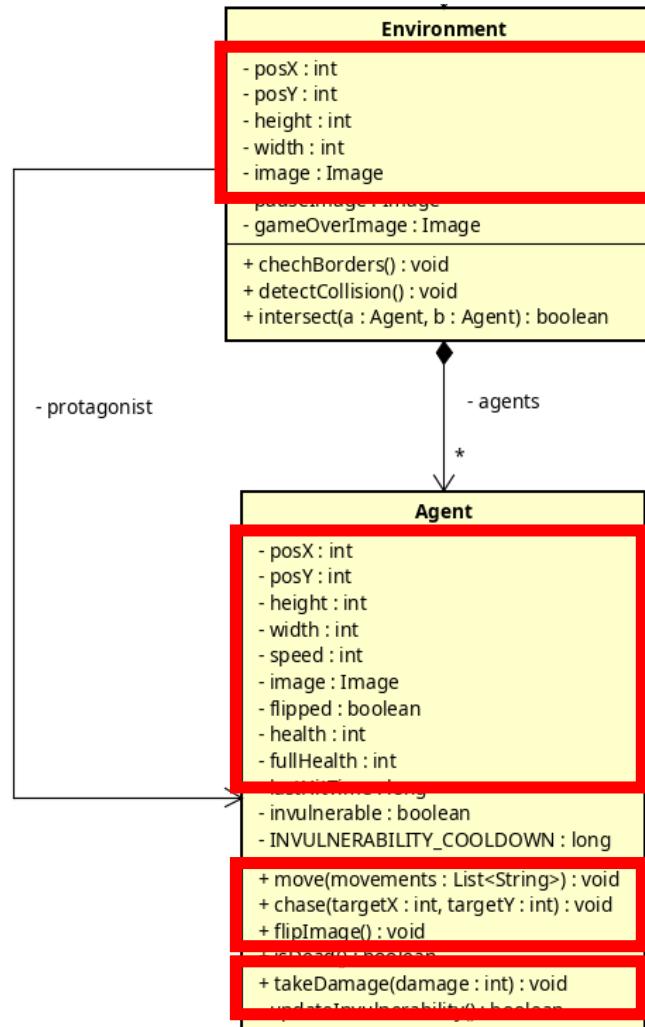
# The Entity



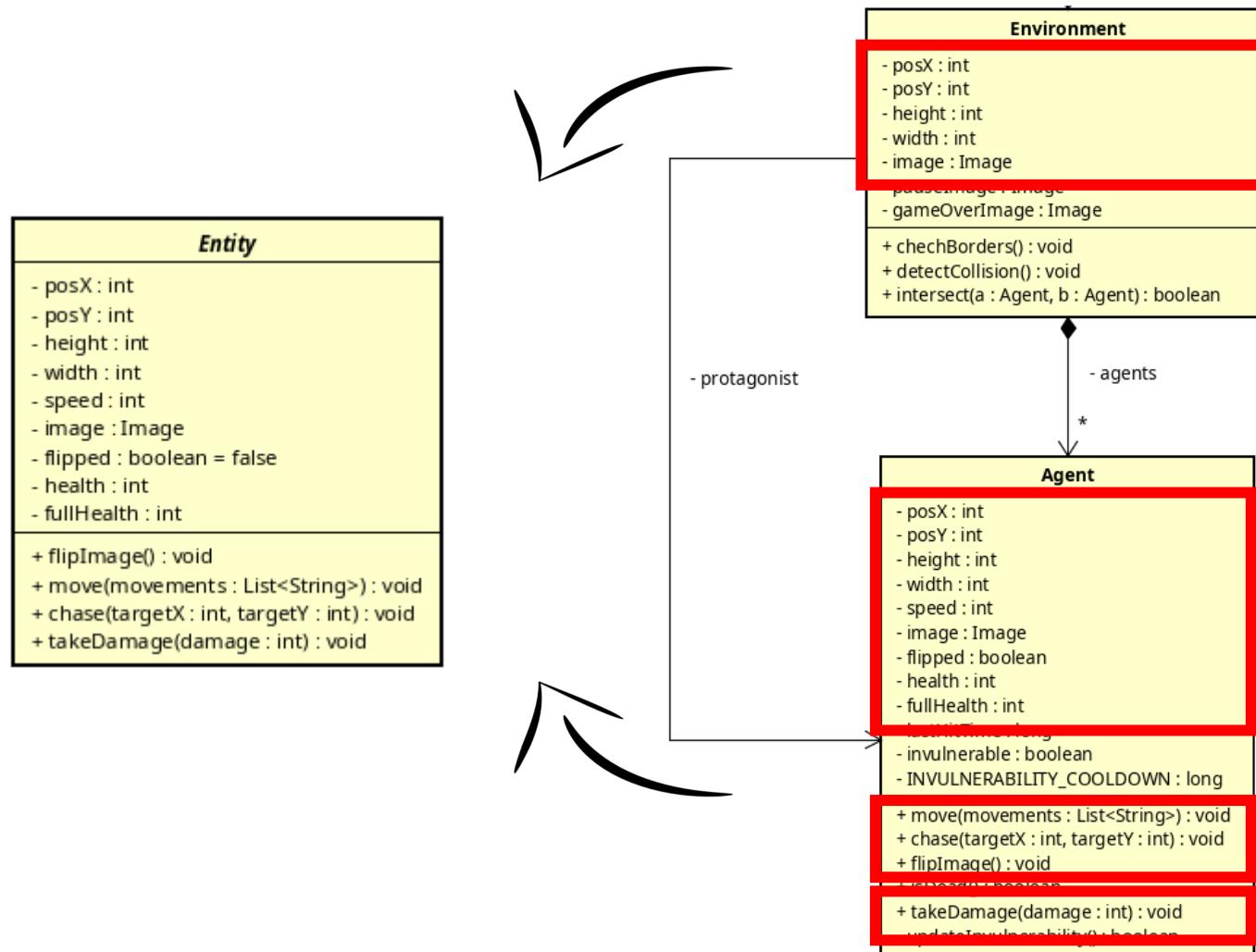
# The Current Model



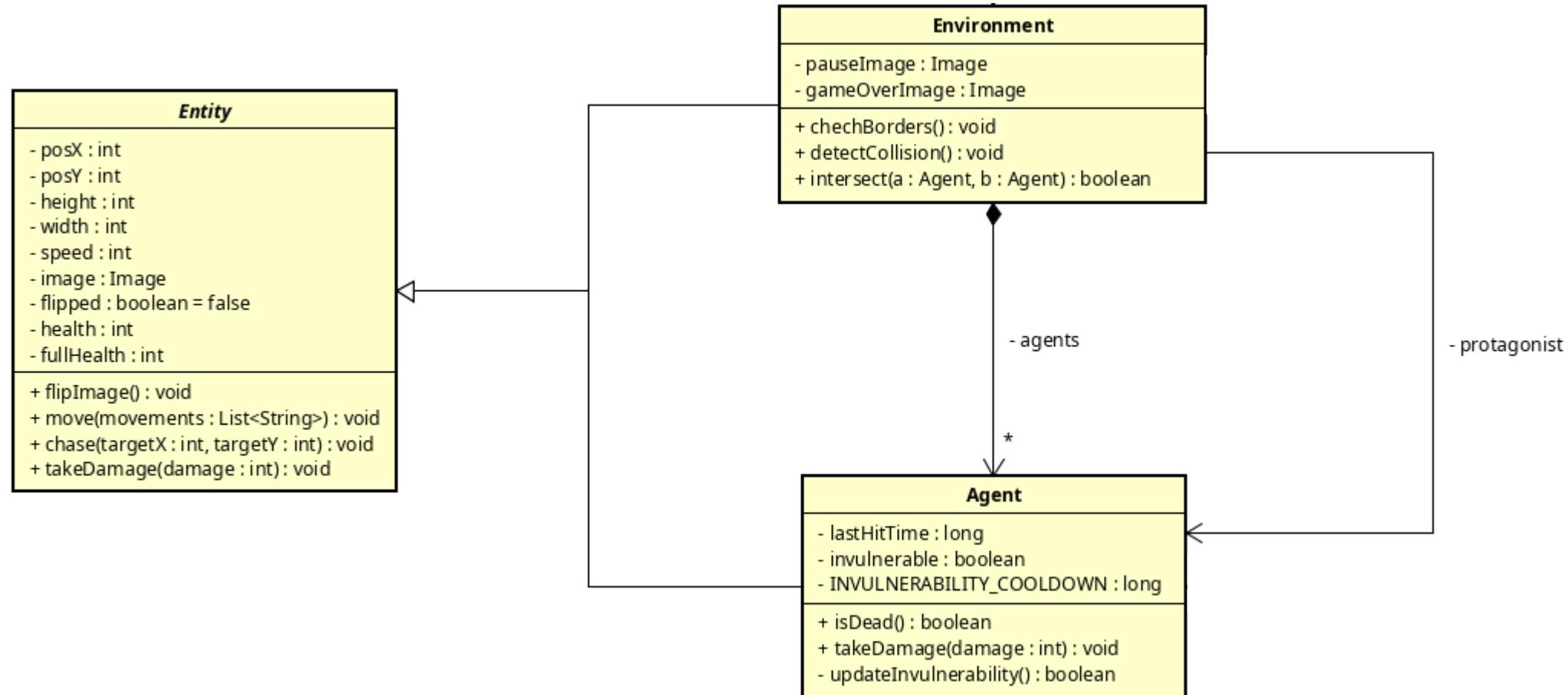
# The Current Model



# New Class Entity



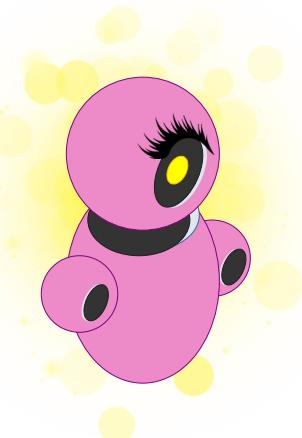
# New Class Entity



# The Entity

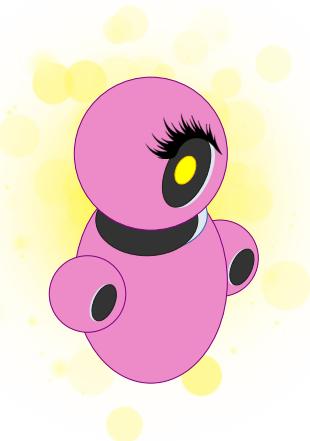
Entities can take damage differently.

# The Entity



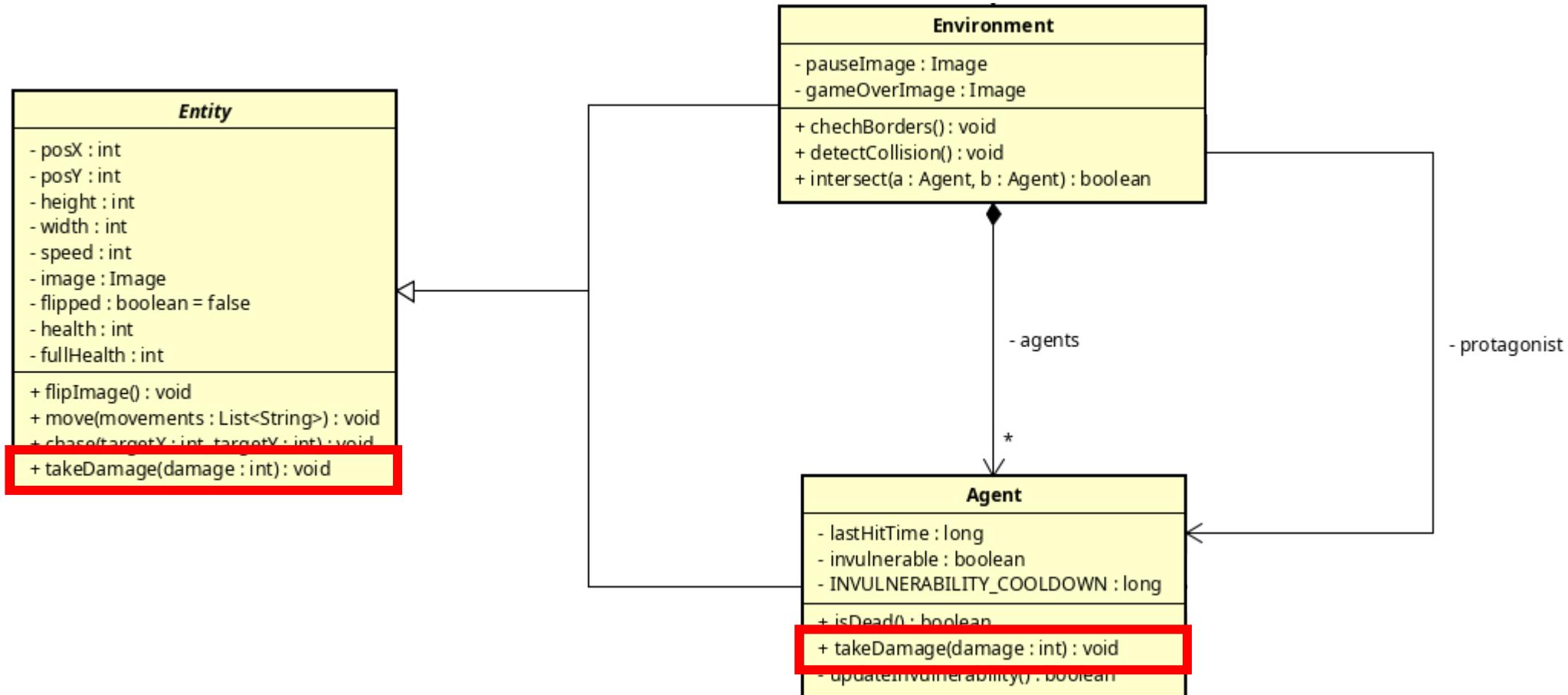
Agents are invulnerable when hit

# The Entity

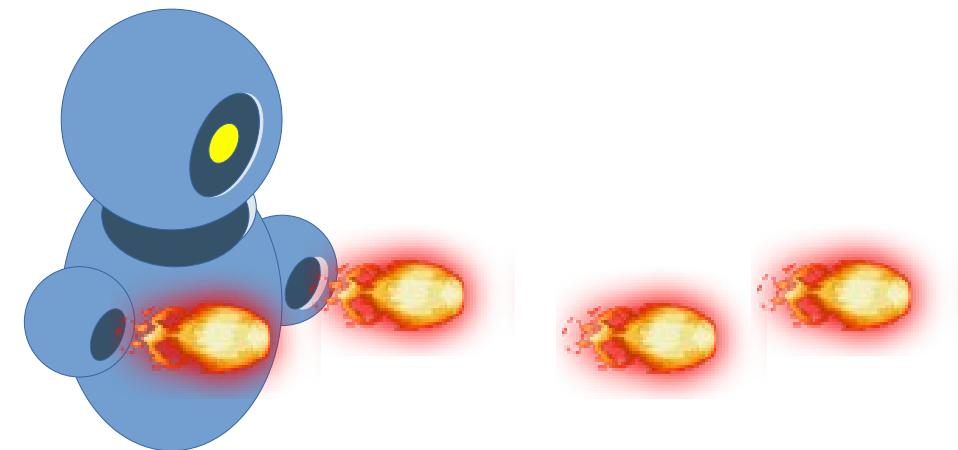


Agents are invulnerable when hit  
other entities **are not**.

# New Class Entity



# SHOOTING WITH A WEAPON



# Shooting

**Shooting** is the ability of **firing projectiles** (such as bullets, arrows, energy blasts, etc.) at targets or opponents.

# Shooting



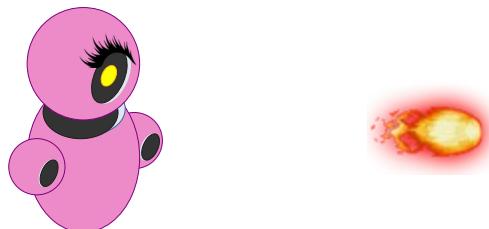
# Shooting

When an agent **fires**...



# Shooting

When an agent **fires**...



# Shooting

When an agent **fires**...



# Shooting

When an agent **fires**...



# Shooting

When an agent **fires**...



... shots are **created**.

# New Classes: Weapon and Shot

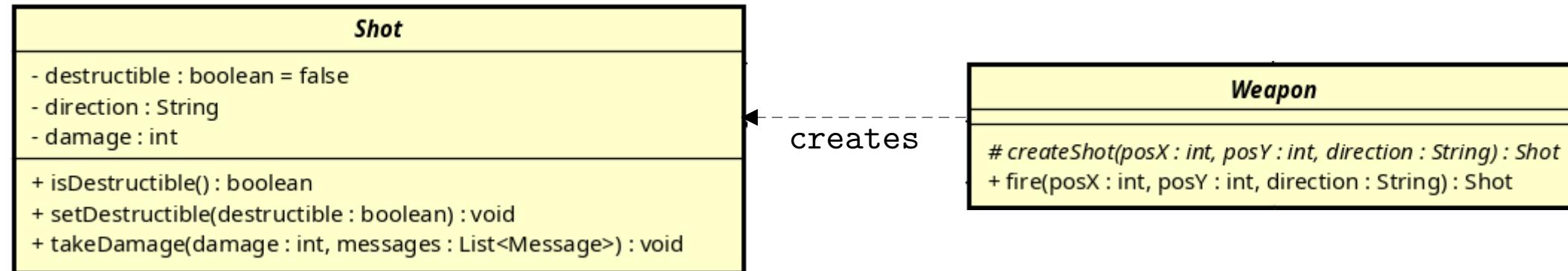
## Shot

```
- destructible : boolean = false  
- direction : String  
- damage : int  
  
+ isDestructible() : boolean  
+ setDestructible(destructible : boolean) : void  
+ takeDamage(damage : int, messages : List<Message>) : void
```

## Weapon

```
# createShot(posX : int, posY : int, direction : String) : Shot  
+ fire(posX : int, posY : int, direction : String) : Shot
```

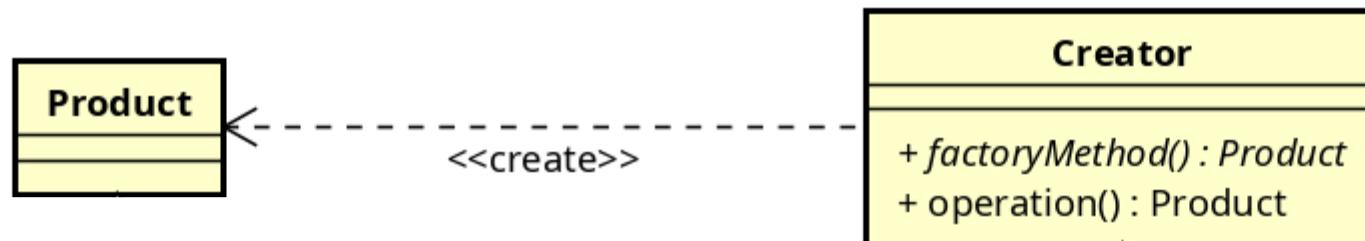
# New Classes: Weapon and Shot



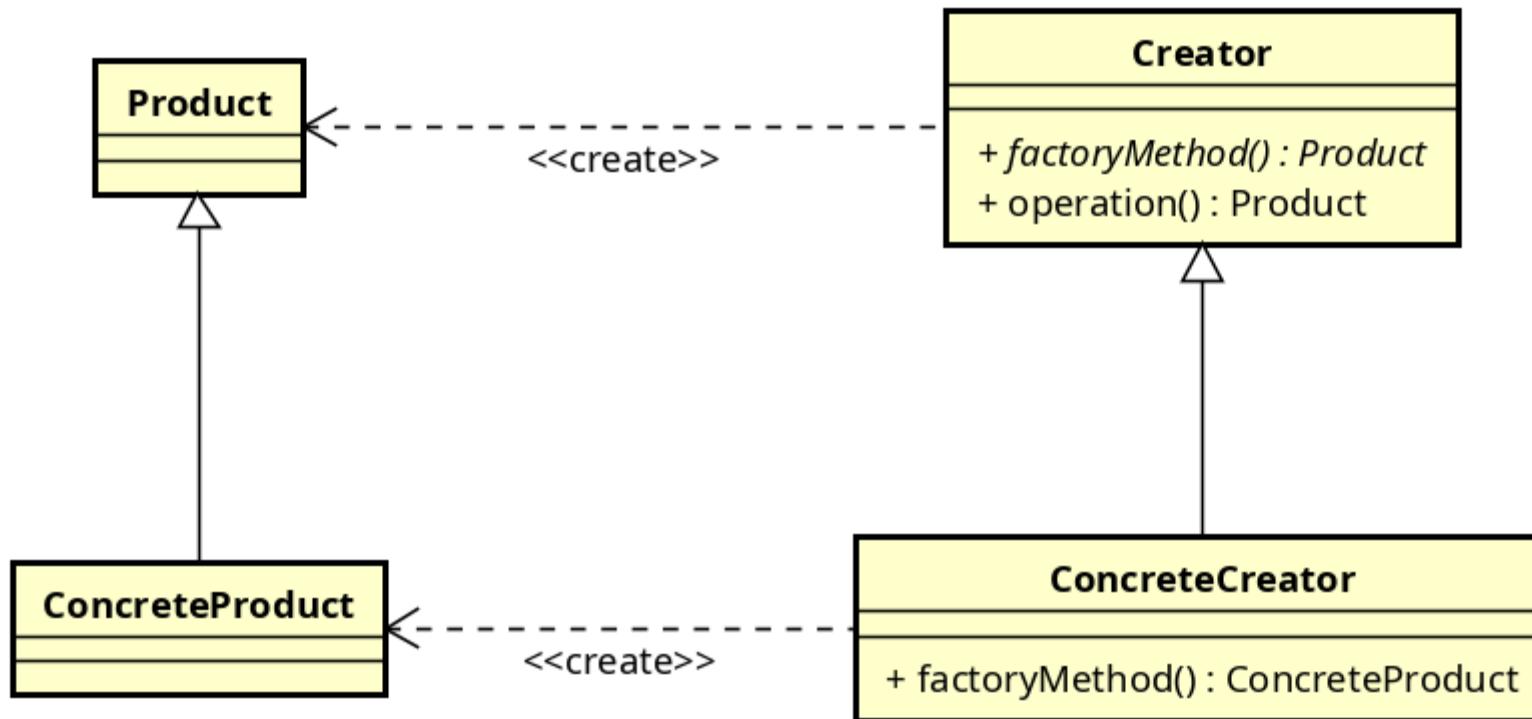
# The Factory Method Pattern

The **Factory Method pattern** is a creational design pattern that defines an interface for creating an object, but lets subclasses decide which concrete class to instantiate.

# The Factory Method Pattern



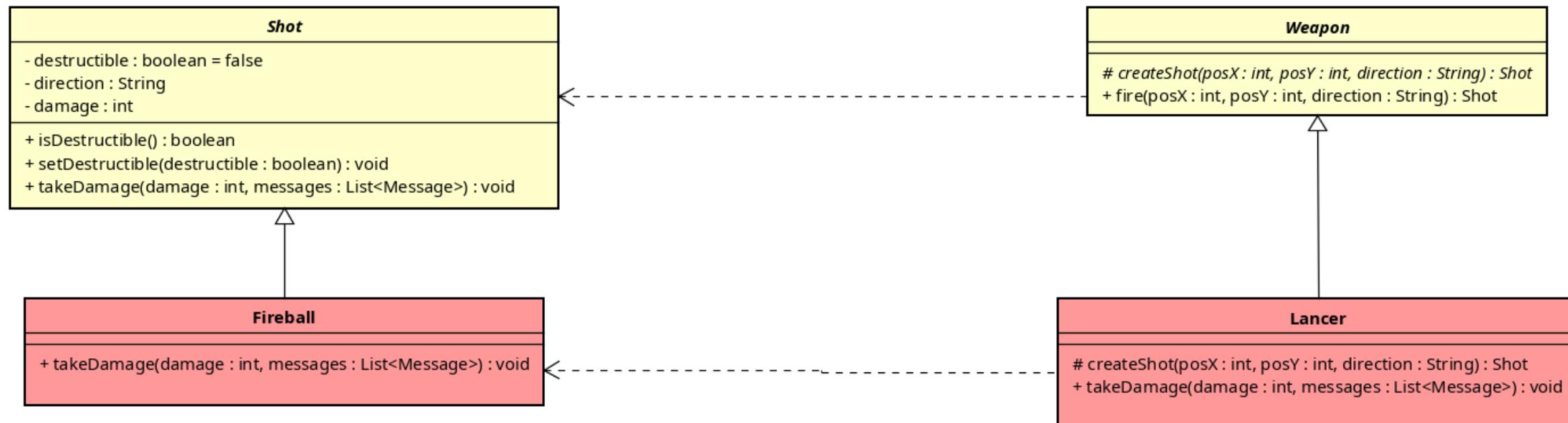
# The Factory Method Pattern



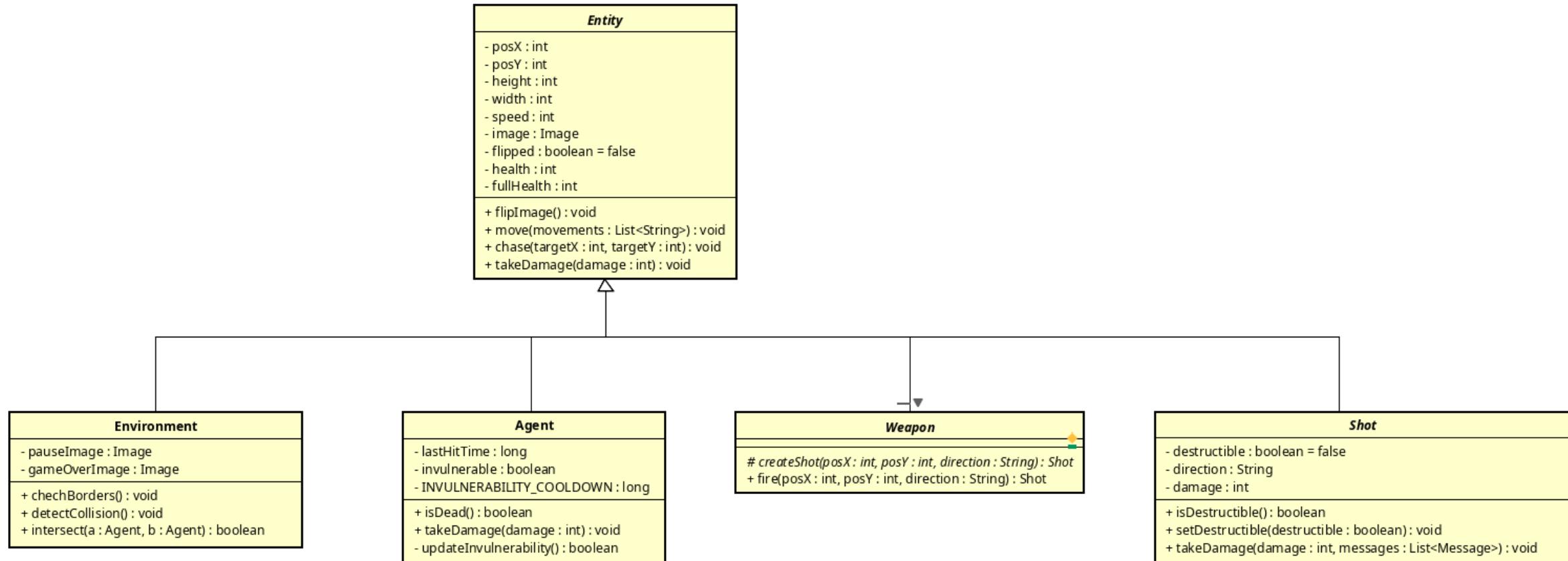
# The Factory Method Pattern



# The Factory Method Pattern



# New Classes: Weapon and Shot



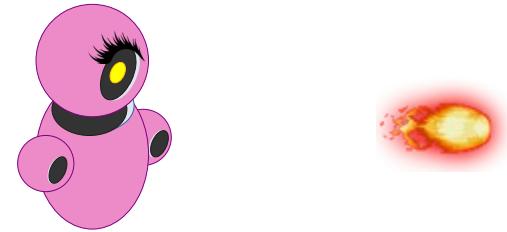
# The Shot Direction

When the agent is NOT flipped...



# The Shot Direction

When the agent is NOT flipped...



... the shot goes to the **RIGHT**.

# The Shot Direction

When the agent is flipped...



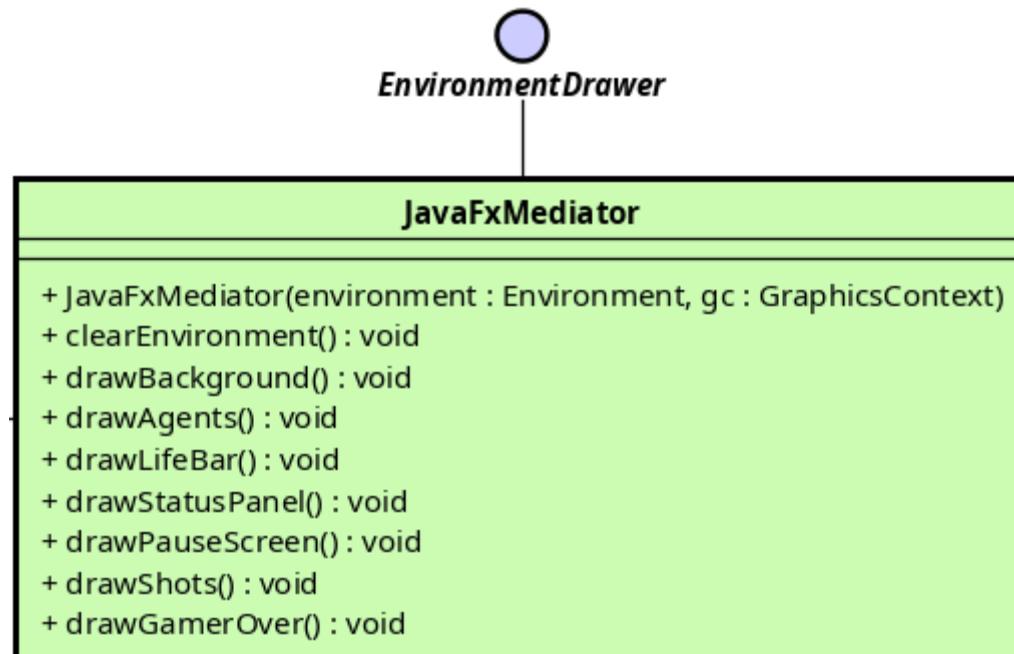
# The Shot Direction

When the agent is flipped...

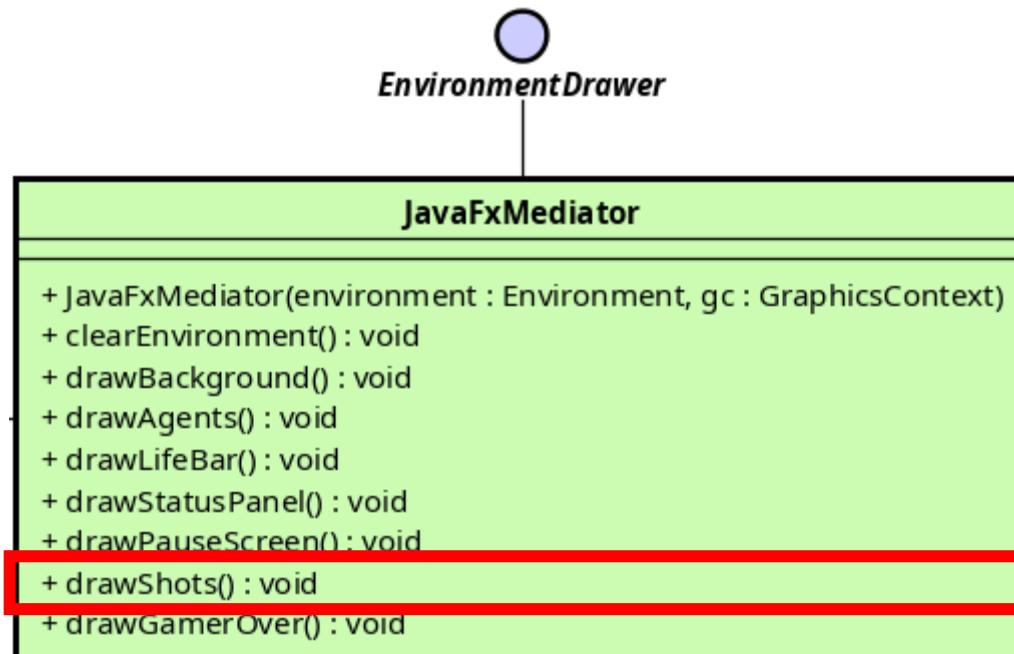


... the shot goes to the LEFT.

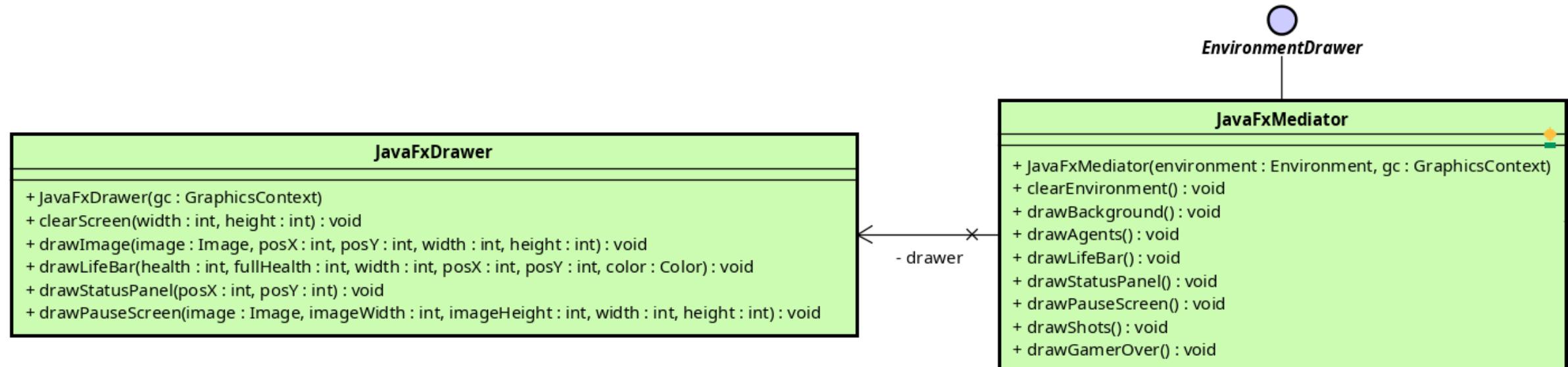
# The Mediator Pattern



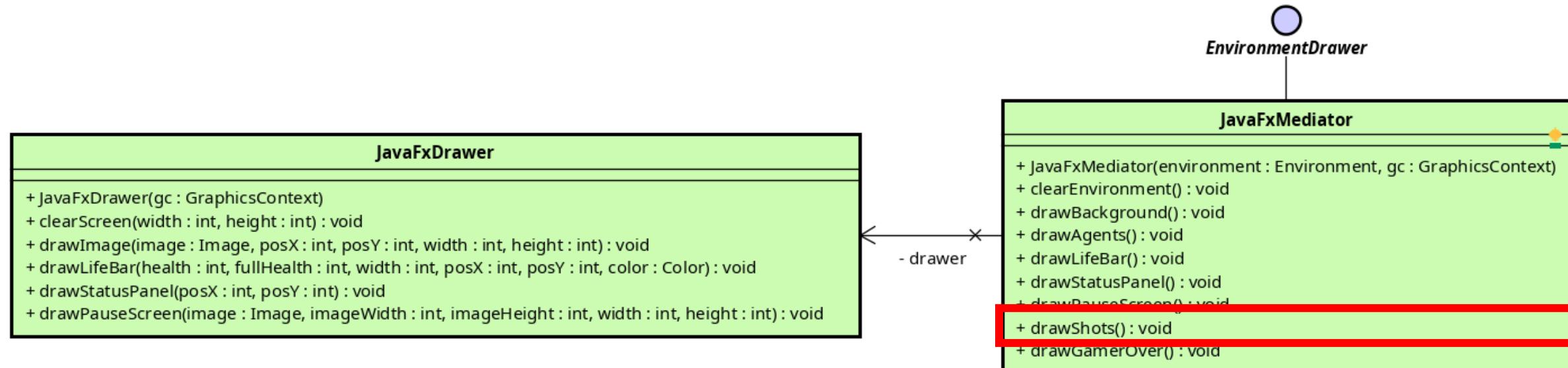
# The Mediator Pattern



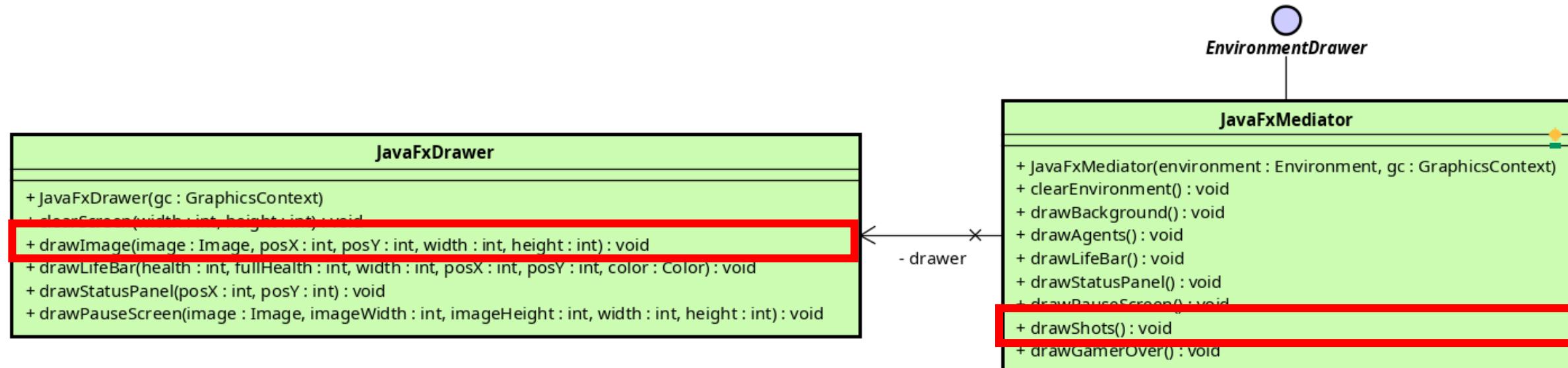
# The Mediator Pattern



# The Mediator Pattern



# The Mediator Pattern



# Limitations

- **The Environment is an Entity.** It has health, for example;
- **It does not show Weapons on screen.** Only Shot instances.