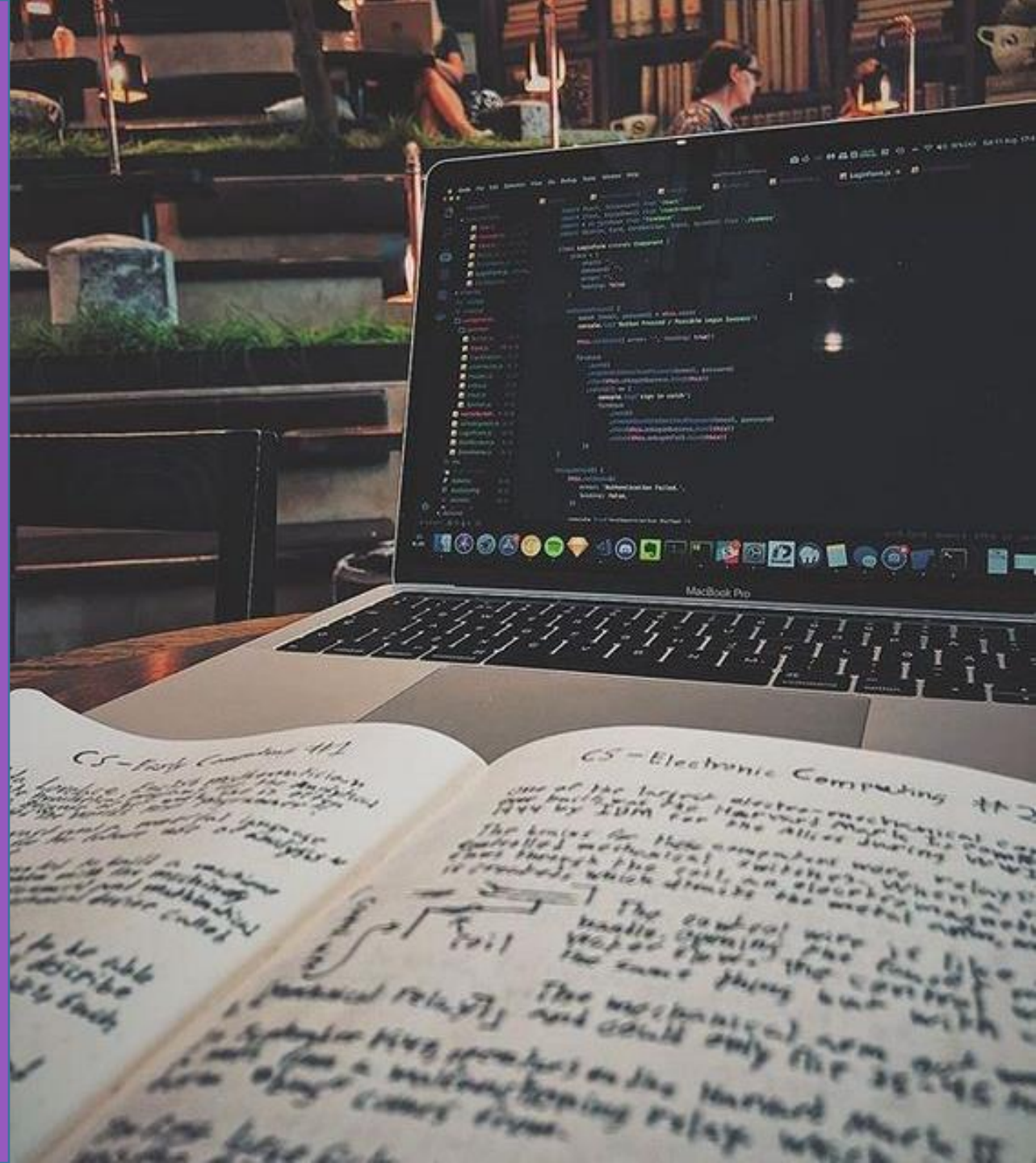
Clase Nro 6

- Clases
- Miembros de una Clase
- Métodos de una Clase
- Visibilidad de los miembros de una clase
- Ejemplo práctico
- Clase estática



Programación orientada a objetos

La programación Orientada a objetos (POO) es un paradigma de programación, que busca crear abstracciones del mundo real a través de objetos.

Clase

La unidad atómica de un sistema orientado a objetos

Permite la creación de objetos.

Encapsula estado y comportamiento

Una estructura de datos

Clase

Declarando una clase

- Declarando una nueva clase que se llama MiClase

```
class MiClase
{
   //miembros y
     atributos de la clase
}
```



```
Typedef struct MiEstructura
{
          //miembros y de la
estructura
}
```

Clase

Creando objetos

- Cada vez que generamos una variable del tipo de la clase decimos que estamos creando un **objeto**
- Cada objeto creado a partir de la clase se denomina **instancia** de la **clase**.
- El operador **new** nos permite reservar memoria de forma dinámica para crear una instancia de la clase

```
MiClase MiInstancia  = new MiClase();
```

Objeto o instancia

Clase

Clase

Campos o atributos

- Llamamos campos de a una variable declarada dentro de una clase.

```
class MiClase
{
    Int x;
    Int y;
}
```

```
Typedef struct MiEstructura
{
    Int x;
    Int y;
}
```

```
MiClase MiInstancia  = new MiClase();
MiInstamcia.x = 10;
```

Clase

Visibilidad de los miembro de una clase

Miembros Públicos y Miembros privados

```
public class Pixel
{
        public Int x;
        public Int y;
}
```

Miembros públicos: Son accesibles desde Afuera de la clase con el operador "."

MiClase MiInstancia = new MiClase();

MiClase.x = 10;

```
public class Pixel
{
        private Int x;
        private Int y;
}
```

Miembros privados: Son accesibles solo desde dentro de la clase

MiClase MiInstancia = new MiClase();

MiClase.x = 10; ← ERROR

No puedes acceder a un miembro privado!

Clase

Propiedades (Getter y Setters)

- Llamamos propiedad a una función que nos permite acceder a las viables declaradas dentro de una clase
- Las propiedades permiten que una clase exponga una manera pública de obtener y establecer valores, a la vez que se oculta el código de implementación o verificación.

```
Pixel NuevoPixel = new Pixel();
NuevoPixel.X = 10;
```

```
public class Pixel
{
        private Int x;

        public int X
        {
                get;
                set;
        }

        private Int y;

        public int y
        {
                get;
                set;
        }
}
```

Propiedades

Propiedades

Clase

Métodos

- Llamamos Método a una función que nos permite acceder a las viables declaradas dentro de una clase.
- Por lo tanto: Un método es un bloque de código que contiene una serie de instrucciones. Un programa hace que se ejecuten las instrucciones al llamar al método.
- Llamar a un método en un objeto es como acceder a un campo. Después del nombre del objeto, agregue un punto, el nombre del método y paréntesis.

```
Pixel NuevoPixel = new Pixel();
NuevoPixel.MoverPunto(10,10);
```

```
public class Pixel
{
        private Int x;
        private Int y;

        public int x
        {
                get;
                set;
        }
        public int y
        {
                get;
                set;
        }

        Public void moverPunto(int Nx, int Ny)
        {
                x = Nx;
                y = Ny;
        }
}
```

Método

Clase

Métodos - Constructor

Un constructor es un método cuyo nombre es igual que el nombre de su tipo. La declaración del método incluye solo el nombre del método y su lista de parámetros; no incluye un tipo de valor devuelto

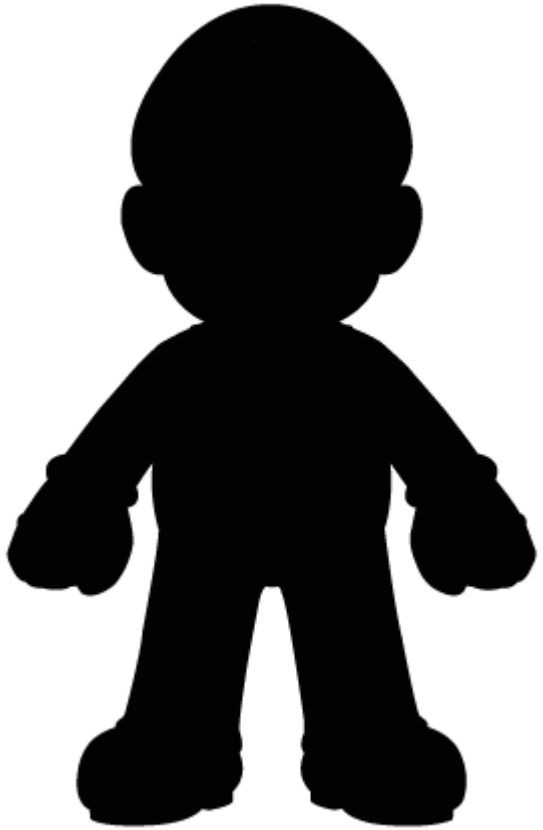
```
Pixel NuevoPixel = new Pixel();
```

Constructor

```
public class Pixel
{
        private Int x;
        private Int y;

        [...]

        Public Pixel()
        {
                x = 0;
                y = 0;
        }

        [...]

}
```

Clase

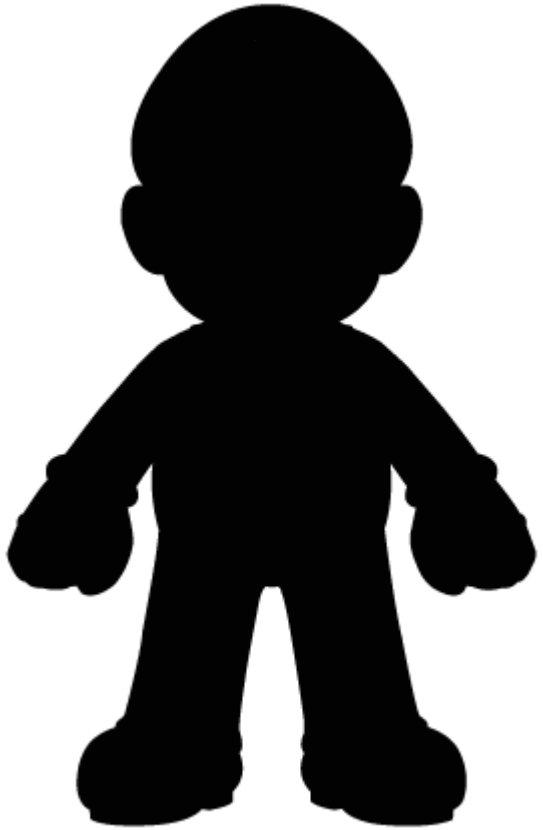
Grupo de elementos de un conjunto que tiene características comunes.



```
public class Player
{
    float PosicionX;
    float PosicionY;
    int vidas

    public void Saltar()
    {
        PosicionX += 5;
        PosicionY += 10;
    }
}
```

Clase

Creando instancias de una clase Player



Player MARIO = *new* **Player**();

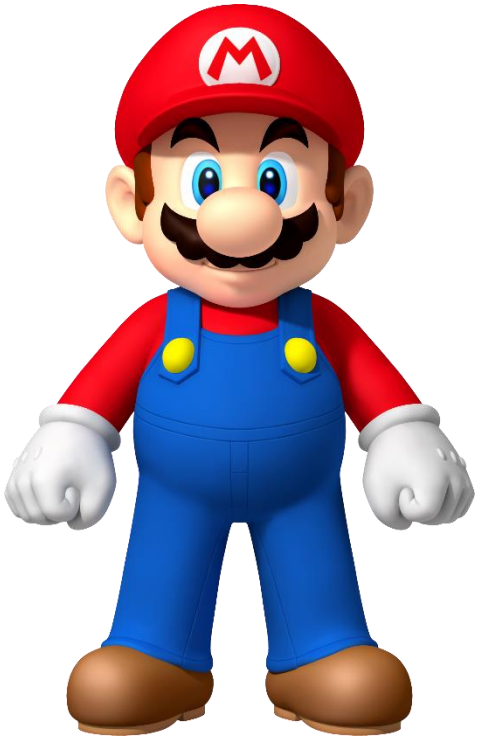
MARIO. PosicionX = 20px;

MARIO. PosicionY = 10px;

MARIO.Vidas = 3;

Clase

Creando instancias de una clase Player



Player MARIO = *new* **Player**();

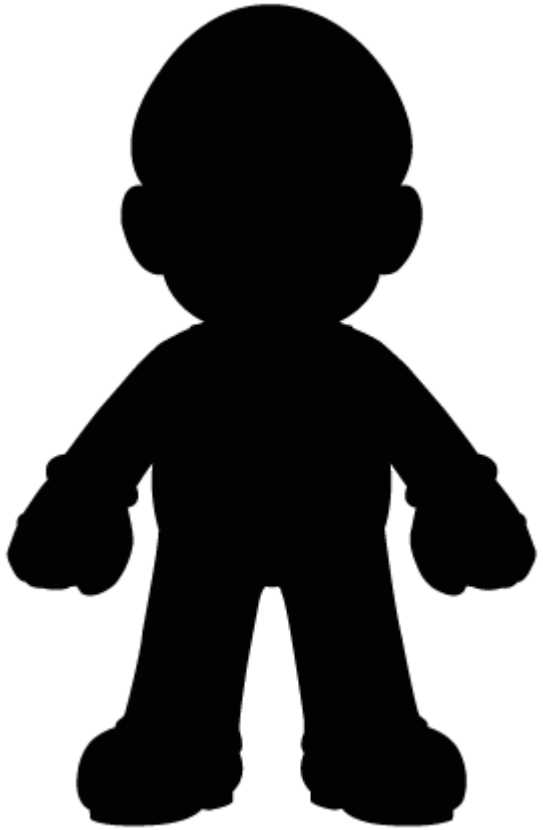
MARIO.PosicionX = 20px;

MARIO.PosicionY = 10px;

MARIO.Vidas = 3;

Clase

Creando instancias de una clase Player



Player LUIGI = *new* **Player**();

LUIGI. PosicionX = 30px;

LUIGI. PosicionY = 10px;

LUIGI.Vidas = 3;

Clase

Creando instancias de una clase Player

Player LUIGI = *new* **Player**();

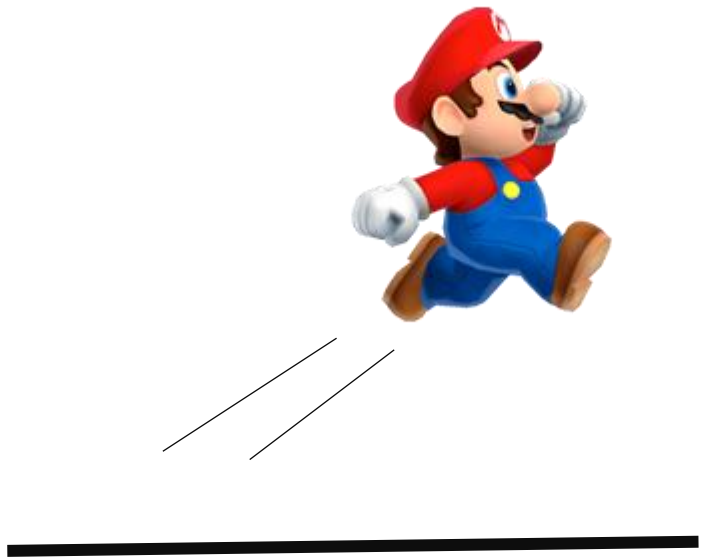
LUIGI. PosicionX = 30px;

LUIGI. PosicionY = 10px;

LUIGI.Vidas = 3;

Clase

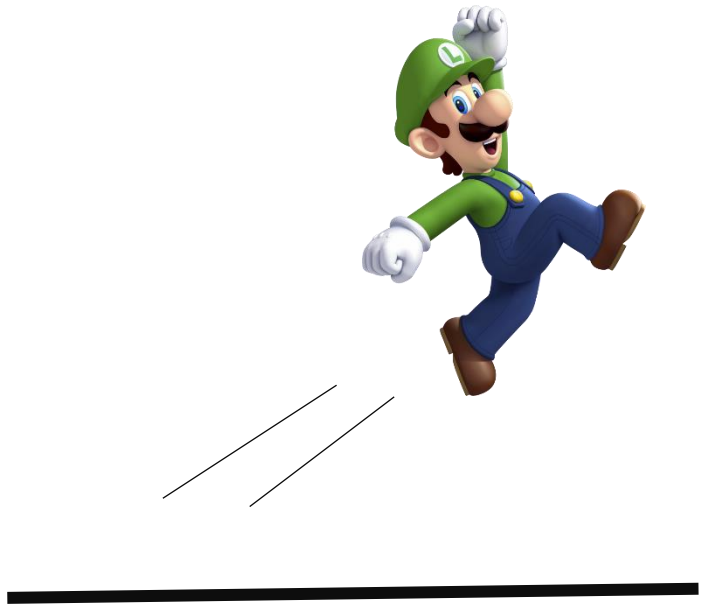
Utilizando un método de la clase Player();



Mario.Saltar();

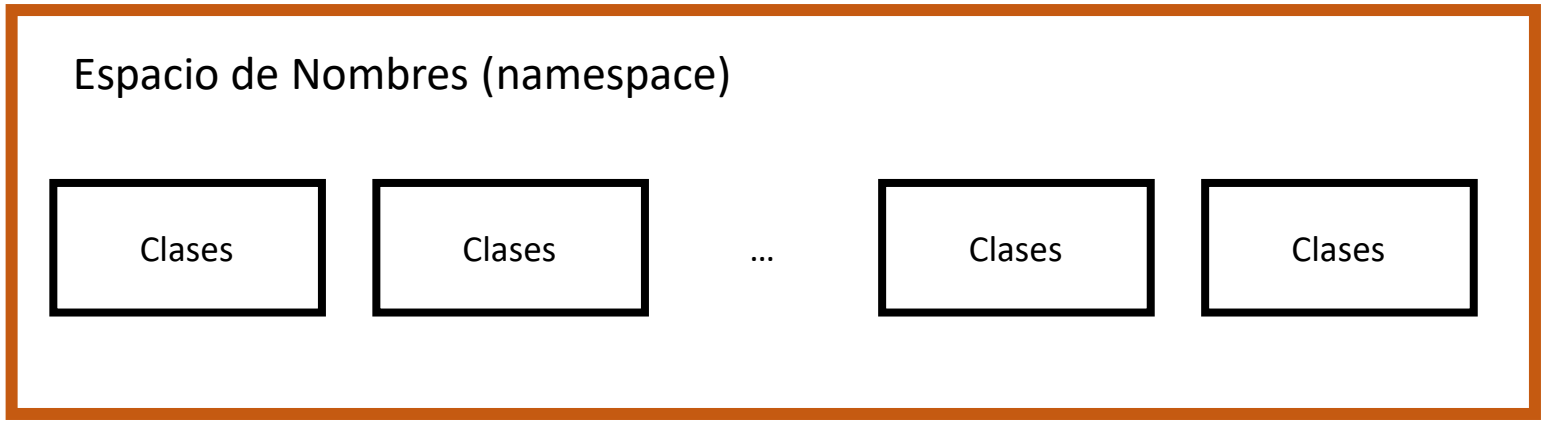
Clase

Utilizando un método de la clase Player();

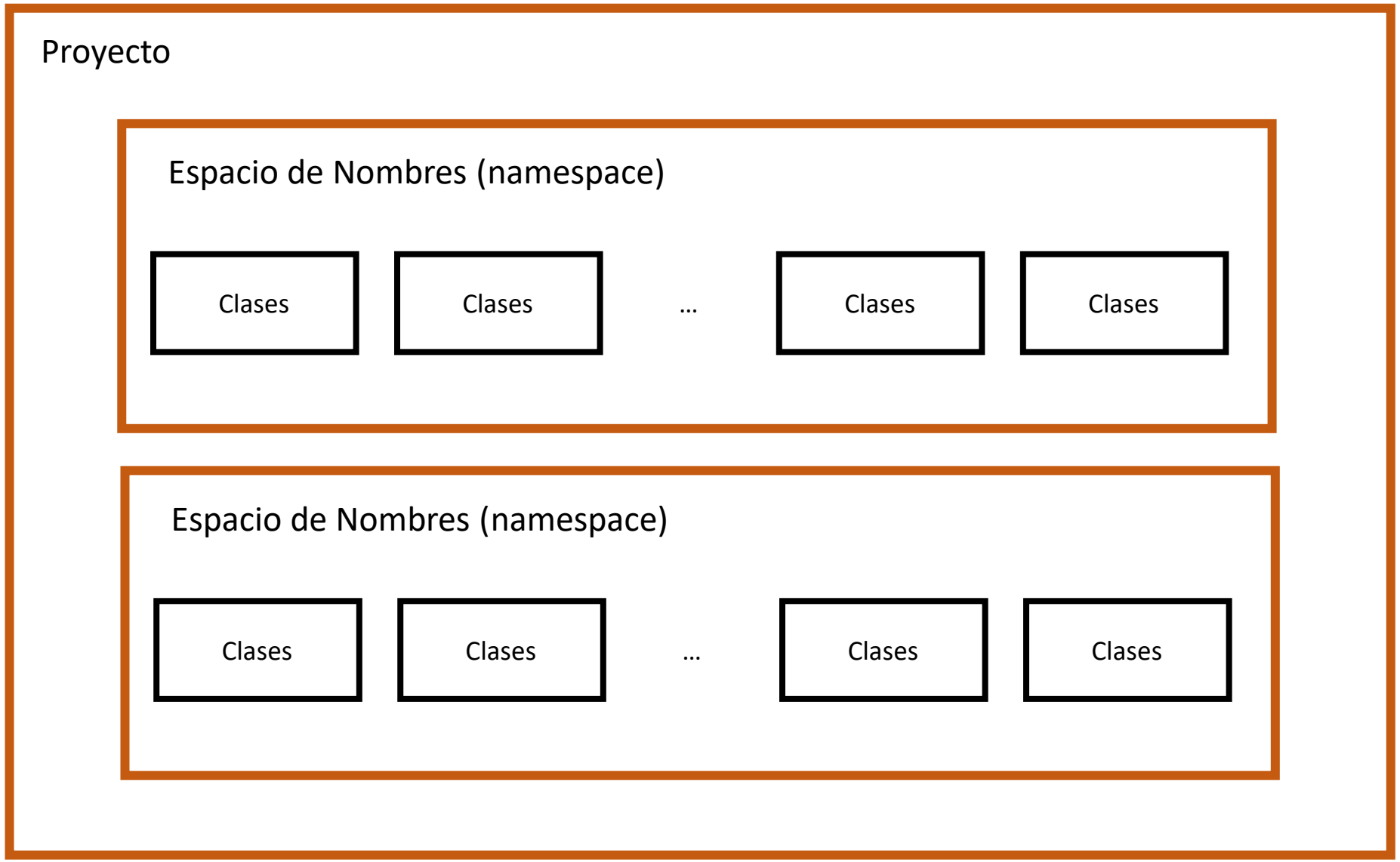


LUIGI.Saltar();

Clase – Organización de las clases



Clase – Organización de las clases



Clase

Las **clases** se utilizan para representar entidades o conceptos, como los sustantivos en el lenguaje. Cada **clase** es un modelo que define un conjunto de variables
-el estado, y métodos apropiados para operar con dichos datos -el comportamiento.

Anatomía de una Clase

Datos – presentado por los campos o atributos

Comportamientos – presentado por los métodos