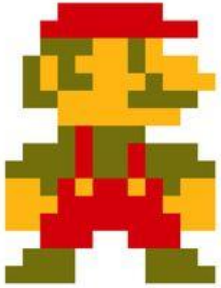


Mario Level X

Ahmad, Connor, Edward

Mario Level 1

- Remake of original mario level 1 using python and pygame

Problems

- Gets boring
- No opportunity for contribution
- No level 2

Solution

- Introduce multi-level support.
- Let players create their own levels

Separate level data from game

Original

This is only for rendering the step components!

```
def setup_steps(self):
    """Create collideable rects for all the steps"""
    step1 = collider.Collider(5745, 495, 40, 44)
    step2 = collider.Collider(5788, 452, 40, 44)
    step3 = collider.Collider(5831, 409, 40, 44)
    step4 = collider.Collider(5874, 366, 40, 176)


    step5 = collider.Collider(6001, 366, 40, 176)
    step6 = collider.Collider(6044, 408, 40, 40)
    step7 = collider.Collider(6087, 452, 40, 40)
    step8 = collider.Collider(6130, 495, 40, 40)

    step9 = collider.Collider(6345, 495, 40, 40)
    step10 = collider.Collider(6388, 452, 40, 40)
    step11 = collider.Collider(6431, 409, 40, 40)
    step12 = collider.Collider(6474, 366, 40, 40)
    step13 = collider.Collider(6517, 366, 40, 176)

    step14 = collider.Collider(6644, 366, 40, 176)
    step15 = collider.Collider(6687, 408, 40, 40)
    step16 = collider.Collider(6728, 452, 40, 40)
    step17 = collider.Collider(6771, 495, 40, 40)

    step18 = collider.Collider(7760, 495, 40, 40)
    step19 = collider.Collider(7803, 452, 40, 40)
    step20 = collider.Collider(7845, 409, 40, 40)
```

1. Solution For Rendering

JSON: Javascript Object Notation

- Easy to understand
- Easy to parse
- Standard

Standard JSON Level Data

```
{
    "data": {
        "pipes": [],
        "steps": [],
        "bricks": [],
        "enemies": [],
        "coin_boxes": []
    },
    "info": {
        "name": "Test Level",
        "author": "test"
    }
}
```

Transfer to JSON

```
def setup_steps(self):
    """Create collideable rects for all the steps"""
    step1 = collider.Collider(5745, 495, 40, 44)
    step2 = collider.Collider(5788, 452, 40, 44)
    step3 = collider.Collider(5831, 409, 40, 44)
    step4 = collider.Collider(5874, 366, 40, 176)
```



```
"steps": [
    {
        "x": 5745,
        "y": 495
    },
    {
        "x": 5788,
        "y": 452
    },
    {
        "x": 5831,
        "y": 409
    },
    {
        "x":58740,
        "y": 366
    }

],
"bricks": [
```

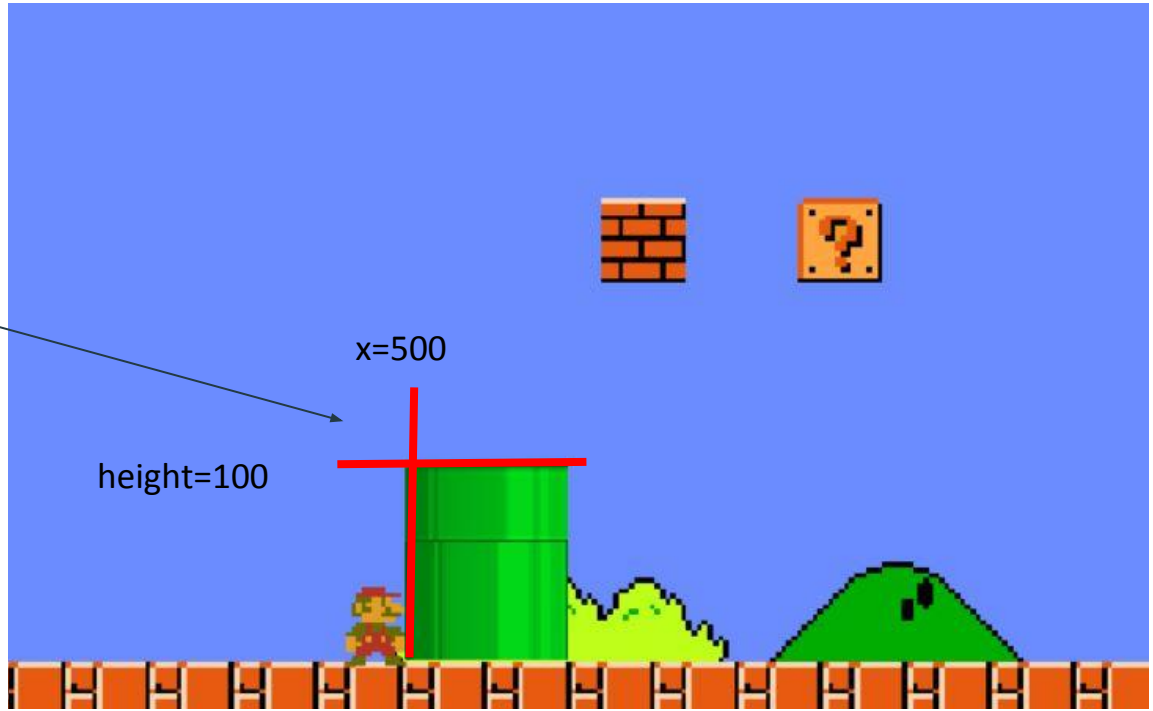
New Method

```
"steps": [
    {
        "x": 5745,
        "y": 495
    },
    {
        "x": 5788,
        "y": 452
    },
    {
        "x": 5831,
        "y": 409
    },
    {
        "x":58740,
        "y": 366
    }

],
```

```
def setup_steps(self):
    """Create collideable rects for all the steps"""
    self.step_group = pg.sprite.Group()
    for step in self.level_data['steps']:
        self.step_group.add(collider.Step(step['x'], step['y']))
```

```
{
    "info": {
        "name": "test",
        "author": "edwardo"
    },
    "data": {
        "enemies": [],
        "pipes": [
            {
                "x": 500,
                "height": 100
            }
        ],
        "steps": [],
        "bricks": [
            {
                "x":600,
                "y":300,
                "contents": "6coins"
            }
        ],
        "coin_boxes": [
            {
                "x": 700,
                "y": 300,
                "contents": "mushroom"
            }
        ]
    }
}
```



2. What about creating levels...

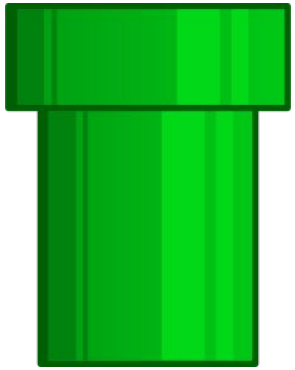
Editor UI

- Keep track of all component objects that were placed.
- Transform map to JSON

Each component should be serializable

Serialize Interface : Pipe Example

pipe = new Pipe(x=10, height = 20)



pipe.serialize()

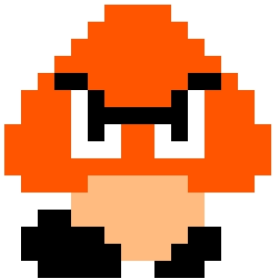
```
def serialize(self):
    return {
        "x": self.rect.x,
        "height": self.rect.y
    }
```

JSON:

```
{
        “x”:10,
        “height”:20
}
```

Serialize Interface : Enemy Example

enemy= new Goomba(x=10, y=20)



enemy.serialize()

```
def serialize(self):
    return {
        'x': self.rect.x,
        'y': self.rect.y,
        'name': self.name
    }
```

JSON:

```
{
        “x”:10,
        “y”:20,
        “name”:”goomba”
}
```

Serialize Map

Simply call serialize() method on all components in the map.

```
def serialize_map(self):

    for component in self.all_components:
        serialized_component = component.serialize()
```

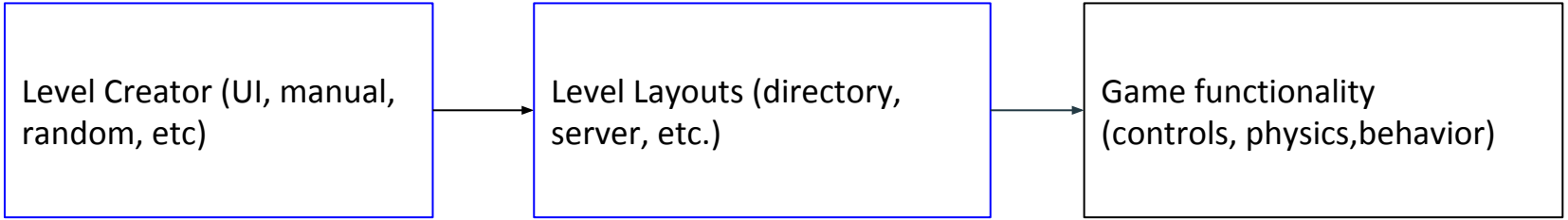
serialize_map()


```
{
    "data": {
        "pipes": [
            {
                "x": 516,
                "height": 323
            }
        ],
        "steps": [
            {
                "x": 344,
                "y": 280
            },
            {
                "x": 430,
                "y": 280
            }
        ],
        "bricks": [
            {
                "x": 258,
                "y": 280,
                "contents": null
            },
            {
                "x": 387,
                "y": 280,
                "contents": null
            }
        ],
        "enemies": [
            {
                "x": 287,
                "y": 240,
                "name": "Goomba"
            },
            {
                "x": 375,
                "y": 220,
                "name": "Koopa"
            }
        ],
        "coin_boxes": [
            {
                "x": 301,
                "y": 280,
                "contents": "mushro
            }
        ]
},
```

```
{
    "data": {
        "pipes": [
            {
                "x": 516,
                "height": 323
            }
        ],
        "steps": [
            {
                "x": 344,
                "y": 280
            },
            {
                "x": 430,
                "y": 280
            }
        ],
        "bricks": [
            {
                "x": 258,
                "y": 280,
                "contents": null
            },
            {
                "x": 387,
                "y": 280,
                "contents": null
            }
        ],
        "enemies": [
            {
                "x": 287,
                "y": 240,
                "name": "Goomba"
            },
            {
                "x": 375,
                "y": 220,
                "name": "Koopa"
            }
        ],
        "coin_boxes": [
            {
                "x": 301,
                "y": 280,
                "contents": "mushro
            }
        ]
    },
```

Render level

Final Model



More:

- Modular
- Scalable
- Customizable
- Fun