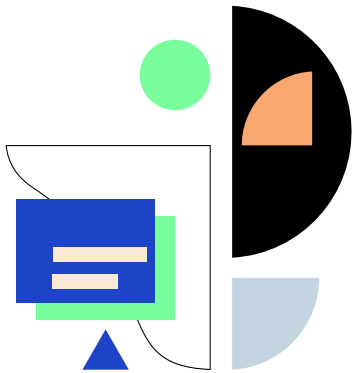
bouvet

MY OLD, SMELLY CODE



Eirik Haaland Bjelland

Statistics

- Method lines:
- 3
- 6
- 6
- 9
- 147
- 163
- 161
- 163
- 59 lines on average
- 1307 lines total in one class

- Code smells found:
- Long method
- Large class
- Primitive obsession
- Comments
- Duplicate code
- Dead code
- Speculative generality
- Divergent change
- Message chains
- Feature envy

bouvet

One class to rule them all

- One class is responsible for almost everything
- A serious case of divergent change

```
public void SortByPosition() {}
public void SortByPositionNew() {}
public void SortByDate() {}
public void SortByDateNew() {}
public void SortByLanguage() {}
public void SortByLanguageNew() {}
public void SortByLanguageBackup() {}
```

```
public void GetAndLoadCubes() {}
public void DeleteAllCubesSafely() {}
public void SphereFilteringSelector(int val) {}
public void HighlightAllSpheres() {}
float ConvertInchesToMetric(string inches) {}
```

```
//Uses cube data to instantiate appropriate amount of cubes, and assigns data to each cube. Also gets data about spheres, and populates the cube with the appropriate spheres.
public void GetCubeRecords()
//Uses pipe data to instantiate appropriate amount of spheres, and assigns data to each sphere
public void GetSphereRecords(string cubeName)
// sends an API request - returns a JSON file. Unused, we need to post, not get.
IEnumerator GetData()
//For posting, but doesn't work with JSON. Unused.
IEnumerator PostData()
IEnumerator SpherePost(string url, string bodyJsonString, string cubeName)
IEnumerator CubePost(string url, string bodyJsonString)
```

Other odours

- Primitive obsession
- Comments
- Duplicate code
- Dead code
- Speculative generality

```
float gap = i * 30f;
newObject.transform.position = new Vector3(12f * u, 0f, -gap + 5f * j);
```

```
//foreach(Sphere sphere in spheres.Reverse())
//{
//    print(sphere.position + " " + sphere.serial);
//    print(cube);
//}

//spheres.Reverse();

//foreach (Sphere sphere in spheres)
//{
//    print(sphere.position + " " + sphere.serial);
//    print(cube);
//}
```

```
public void SortByLanguageBackup() {}
// sends an API request - returns a JSON file. Unused, we need to post, not get.
IEnumerator GetData() { }
//For posting, but doesn't work with JSON. Unused.
IEnumerator PostData() { }
```

More wicked wafts

- Feature envy

- Message chain

```
public void SortByPositionNew()
{
    List<Sphere> spheres = FindObjectsOfType<Sphere>().ToList();

    List<List<Sphere>> sphereSize2DArr = new List<List<Sphere>>();

    List<string> uniqueSphereSizes = new List<string>();
    for (int x = 0; x < spheres.Count; x++)
    {
        if (!uniqueSphereSizes.Contains(spheres[x].radius))
        {
            uniqueSphereSizes.Add(spheres[x].radius);
        }
    }
}
```

```
case "Red":
    {
        Color colorRed;

        ColorUtility.TryParseHtmlString("#600901", out colorRed);
        GetComponent<Renderer>().material.color = colorRed;
        break;
    }
```

```
sphere.name = "Sphere " + uniqueSphereSizes[uniqueSphereSizesMetric[i].Value];
sphere.GetComponentInChildren<TextMeshProUGUI>().text = uniqueSphereSizes[uniqueSphereSizesMetric[i].Value];
sphere.transform.SetParent(newCube.transform);
```

In conclusion

- Had never heard of code smells, SOLID, or any other programming principles
- Needed to get things to work relatively quickly
- Customer was happy!

bouvet

bouvet

Thank you!

Questions?

Eirik Haaland Bjelland

eirik.bjelland@bouvet.no

References

- https://refactoring.guru/refactoring/smells