

I'LL JUST USE MY SHOTGUN

A talk about code smells

Jon Oftedal Høberg



bouvet



What are code smells?

- Not bugs or errors
- Why is it important to nip them in the bud?
- Maintainability
- Readability
- Extensibility
- Prevent future bugs



Types of code smells

- Bloaters
- Object-Orientation abusers
- Dispensables
- Couplers
- Change preventers

Shotgun surgery

- Make a small change?
- Think again buddy



8 references | Jon Oftedal Høberg, 20 days ago | 1 author, 1 change | 1 work item

```
public string UserUniqueAzureId { get; set; }
```



8 references | Jon Oftedal Høberg, 20 days ago | 1 author, 1 change | 1 work item

```
public Guid UserUniqueAzureId { get; set; }
```



- ❌ CS0029 Cannot implicitly convert type 'string' to 'System.Guid'
- ❌ CS1503 Argument 3: cannot convert from 'string' to 'System.Guid'
- ❌ CS0029 Cannot implicitly convert type 'System.Guid' to 'string'
- ❌ CS1503 Argument 1: cannot convert from 'string' to 'System.Guid?'
- ❌ CS1503 Argument 1: cannot convert from 'string' to 'System.Guid?'
- ❌ CS1503 Argument 3: cannot convert from 'string' to 'System.Guid'
- ❌ CS0019 Operator '==' cannot be applied to operands of type 'string' and 'Guid'



Adding a role

```

04 references | Mats, 40 days ago | 2 authors, 2 changes | 2 work items
public interface IMeetingAuthorizationHelper
{
    29 references | Mats, 93 days ago | 1 author, 1 change
    Task<bool> UserIsProjectInformation(ClaimsPrincipal user, Guid? projectId);
    1 reference | 0 changes | 0 authors, 0 changes
    bool UserIsNonGuestParticipantInMeeting(ClaimsPrincipal user, GeneralMeeting meeting);
    16 references | 0 changes | 0 authors, 0 changes
    Task<bool> CheckFutureMeetingAccess(ClaimsPrincipal user, FutureMeetingResource resource);
    35 references | 0 changes | 0 authors, 0 changes
    Task<bool> CheckReadAccessAsync(ClaimsPrincipal user, BaseMeetingDescriptor meetingDescriptor);
    9 references | 0 changes | 0 authors, 0 changes
}

```

```

6 references | - changes | -author
private Func<GeneralMeeting, bool> UserHasAccess(Expression<bool> accessExpression)
{
}

```

```

1 reference | XXXXXXXXXX 38 days ago | 1 author, 1 change | 1 work item
private static Expression<Func<GeneralMeeting, bool>> CreateUserAccessExpression(

```

So what can we do?

- Move methods and fields
- Organize
- Maximize cohesion
- Minimize coupling
- Reduce code duplication.



QUESTIONS?

Jon Oftedal Høberg

jon.hoberg@bouvet.no



Sources

- Alcor Academy
- Refactoring guru
- <https://refactoring.guru/>