Access Control with Expressions

Bryan Hansen
twitter: bh5k
http://www.linkedin.com/in/hansenbryan

pluralsight
hardcore dev and IT training

Expressions

```
<http auto-config="true" use-expressions="true">
    <intercept-url pattern="/admin*" access="hasRole('ROLE_ADMIN')" />
</http>
```

use-expressions="true"

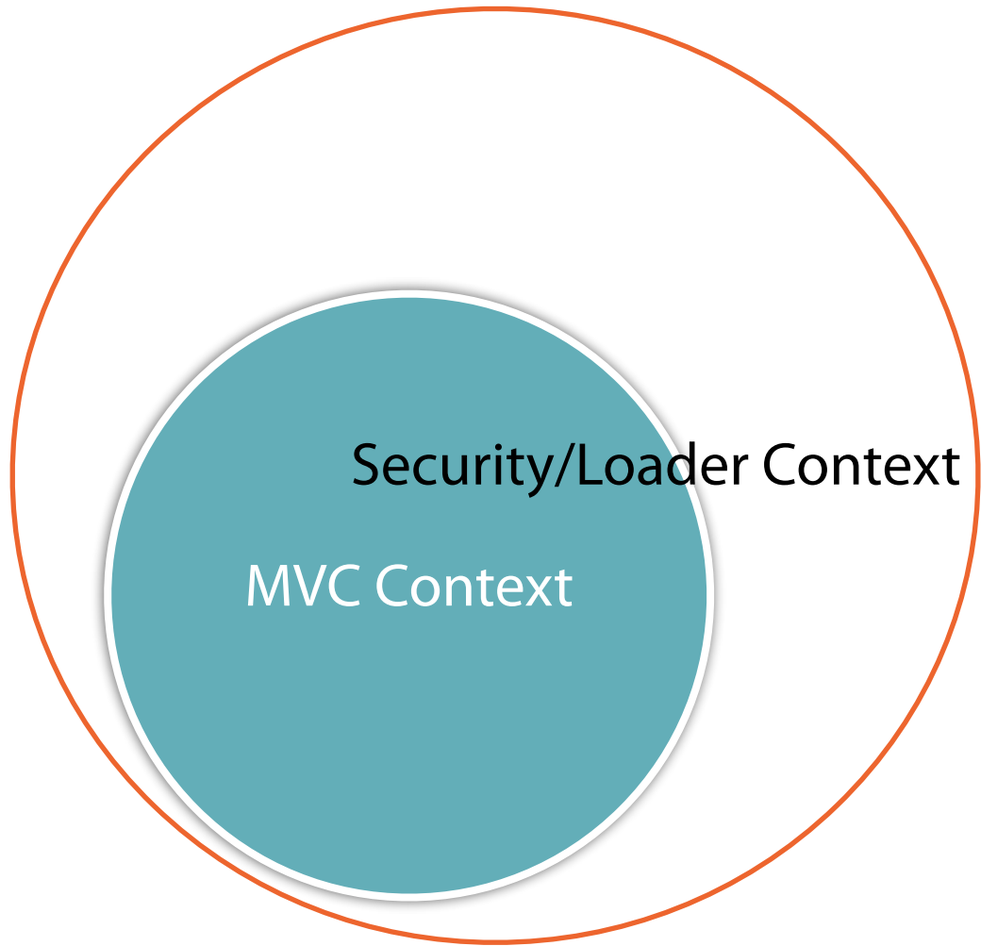
- Attribute of http element
- Simplify boolean logic into an expression
- Built-in expressions:
  - hasRole
  - hasAnyRole
  - permitAll
  - hasPermission
- All or nothing

Method Level Security

- Uses the same expressions
- @PreAuthorize - most common
- @PostAuthorize - ran after method executes
- <global-method-security />
  - pre-post-annotations
  - secured-annotations
  - jsr250-annotations
- Context matters

Spring Context

- We have two contexts in our app
  - mvc context
    - loaded by servlet
  - security (and everything else) context
    - loaded by loader listener



Permissions

- Able to enable per object permissions
  - A level deeper than ROLE
- hasPermission(#goal, ‘createGoal’)
  - Tied to an object, #goal
  - Permission, createGoal
- <security:expression-handler ref=“fitnessExpressionHandler” />
- Custom Permission Evaluator
  - FitnessPermissionEvaluator

Permission Evaluator

- Interface with two methods:
  - hasPermission(Authentication auth, Object targetObj, Object permission)
  - hasPermission(Authentication auth, Serializable id, String type, Object permission)
- Every implementation will be different
  - Inject whatever you need to the evaluator

Summary

- Expressions
- Method Level Security
- Domain Level Permissions