# JAV201 - DEFENDING JAVA

## Course Learning Objectives

The eight modules of this course focus on widely accepted practices for protecting Java applications. These include implementing authentication and authorization, using trusted security frameworks, validating data and storing it safely, creating servlet filters to help secure input validation, conversion, logging, compression, encryption, and decryption, ensuring that files are handled securely, preventing attacks, implementing cryptography, managing secrets, using certificates and key stores, signing and timestamping code, and comparing how different logging frameworks handle security.

## Description

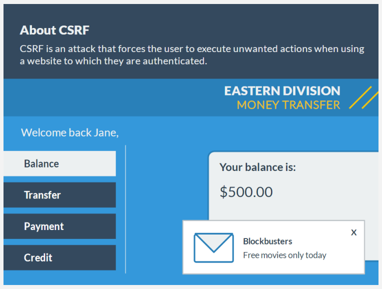
Defending Java is for Java developers and Java architects. This course focuses on best practices for addressing threats against Java applications. Suggested prerequisite courses include Defending Web Applications, OWASP Top 10, and Secure Software Design.

## Audience

Java Developers
Java  Architects

## Time Required

Tailored learning - 120 minutes total



About CSRF

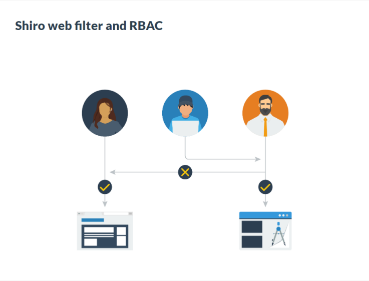
CSRF is an attack that forces the user to execute unwanted actions when using a website to which they are authenticated.

Welcome back Jane,

Your balance is:

$500.00

Blockbusters
Free movies only today



Shiro web filter and RBAC

Variety of parsers

For more information on XXE Prevention:

# JAV201 - DEFENDING JAVA

## Course Outline

### 1. Authentication and Authorization

- Common vulnerabilities
- Authentication and authorization types
- Code: Apache Shiro
- Shiro web filter and RBAC
- Spring Security
- Code: Spring Security
- Comparing Apache Shiro and Spring Security

### 2. Web Defenses

- Introductiont to web defenses
- Securing cookies
- Cookie hijacking
- Cookie hijacking defense
- Server-side sessions
- Session timeouts
- Session max timeout
- Session max age
- Session fixation
- Servlet filters
- Servlet filter implementation
- Functional endpoint
- Testing the implementation
- Cross-Site Request Forgery
- REST endpoints
- Token system
- HTTP request referrers
- Cross-Origin Resource Sharing
- Code: Cross-Origin Resource Sharing
- Servlet filters

### 3. Data Validation

- The importance of input validation
- Input validation strategies
- Email address validation
- Allowlists and denylists
- Code: regex based allowlist
- Open-source frameworks
- SQL injection
- SQL injection defense
- Resource injection
- Reflected XSS
- Stored XSS
- Defense against XXS
- Deserialization attacks
- Defense against deserialization attacks
- Code: Defense against deserialization attacks
- Code: JEP 290
- HTTP response splitting
- Code: Using a denylist
- Code: Using regular expressions
- Upload verification
- Special file validation
- Image upload verification

### 4. File Handling

- Java file APIs
- Local file inclusion
- Poison Null Byte
- Indirect selection
- Polyglot VM
- Marshalling and unmarshalling
- Memory overflow
- XML parsers
- Parsers
- Variety of parsers
- Enforcing file size
- File size defense
- File access blocking

### 5. Cryptography

- JCA file APIs
- Random numbers
- Locations of core cryptography
- Symmetric encryption driver
- Encrypt function
- Decrypt function
- Symmetric encryption test
- Asymmetric encryption
- Asymmetric encryption driver
- Encrypt function
- Decrypt function
- Asymmetric encryption test
- Benefits of cipher streams
- Wrapping a stream
- Chained streams

### 6. Secrets Management

- Certificates
- Java certificate management
- Diagnosing certificate issues
- Key locations
- Keystores
- Listing aliases
- Keytool
- Generating keys
- Signed code
- Validating code
- Signing JARs
- Timestamped signatures

Security Compass

# JAV201 - DEFENDING JAVA

## Course Outline

### 7. Logging and Auditing

- Java logging
- Logging frameworks
- System.Logger Interface
- Code: System.Logger Interface
- Pluggable logging framework
- How logging works
- Logging levels
- Logging filters
- Centralized logs
- Unified log frameworks
- Logging sensitive information
- Handling sensitive information

### 8. Best Security Practices

- Update all security releases
- Use tried and tested libraries
- Scan for open-source vulnerabilities
- Deserialization vulnerabilities
- Defense against deserialization vulnerabilities
- Race conditions
- Race conditions in Java web applications
- Race condition example
- Eliminating race conditions

Security Compass