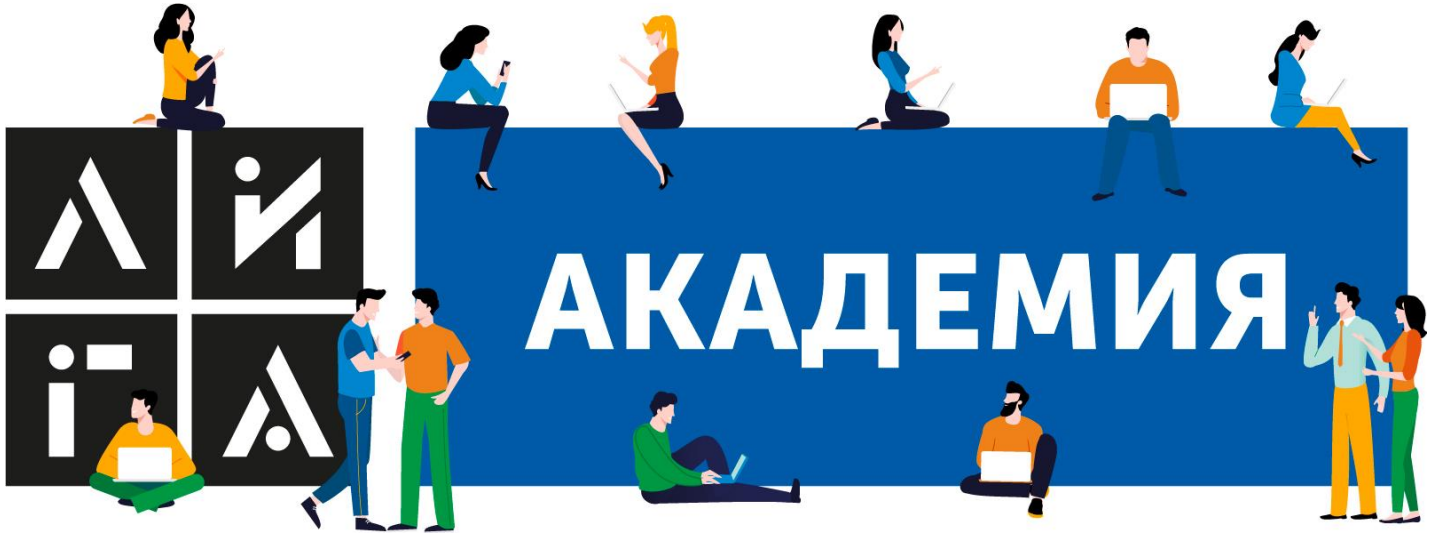

АКАДЕМИЯ

Безопасность приложений

Spring Security и прочая крипта

01.

Основные понятия

02.

Базовая криптография

03.

Spring Security

04.

JSON Web Token

АКАДЕМИЯ

Основные понятия

Аутентификация VS идентификация

АКАДЕМИЯ

Основные понятия

Идентификация

- Процесс взаимодействия с системой, в рамках которого субъект сообщает свой идентификатор в рамках системы
- Примеры: ФИО, логин, табельный номер, номер паспорта и т.д.

Аутентификация

- Процесс, в рамках которого идентифицированный субъект подтверждает подлинность идентификации
- Примеры: ввод пароля, считывание отпечатка пальцев, ввод ответа на секретный вопрос, ввод кода из СМС, присланного на доверенный телефон

Авторизация

- Проверка прав аутентифицированного субъекта на совершение операции над защищаемым объектом.
- Примеры: проверка прав на чтение или запись файла, проверка прав на доступ к web-странице, проверка прав на оплату картой

Основные понятия

Симметричное шифрование

- Симметричным называется шифрование, при котором для шифрования и дешифрации используется один и тот же ключ
- Используется в тех случаях, когда ключ не надо передавать сторонним лицам или есть возможность безопасно передать копию ключа

Асимметричное шифрование

- Асимметричным называется шифрование, при котором для шифрования и дешифрации используются разные ключи – для шифрования используется открытый ключ, принадлежащий адресату, а для дешифрации адресат использует свой секретный ключ, который неразрывно связан с публичным ключом, использованным для шифрации
- Используется в том случае, если необходимо шифровать информацию для сторонних лиц и при невозможности обеспечить безопасную передачу ключа шифрования

Электронная подпись

- Электронная подпись – генерация массива информации, которая может однозначно подтвердить, что для формирования подписи использовался секретный ключ автора подписи. Проверить подпись можно открытым ключом, а сформировать только закрытым.
- Используется при необходимости однозначно удостоверить согласие владельца ключа на совершение операции. Это могут быть согласование документа, вход по ЭП и т.д.

Базовая криптография

BCrypt и java.security

АКАДЕМИЯ

Популярные библиотеки

bcrypt

- Простой и достаточно надежный алгоритм хэширования паролей
- Имплементируется библиотекой jBCrypt
- **Пример**: **String hashed = BCrypt.hashpw(password, BCrypt.gensalt(12));**

Java.security

- Библиотека функций безопасности с широкими возможностями
- Обеспечивает криптографию, подписывание, аутентификацию, безопасные соединения, контроль доступа
- Входит в состав стандартной библиотеки
- Пример хеширования:
  MessageDigest md = MessageDigest.getInstance("SHA-1");
  byte[] hashedPassword = md.digest("password".getBytes());
- Пример генерации сертификата:
  SecureRandom secureRandom = new SecureRandom();
- **KeyPairGenerator keyPairGenerator = KeyPairGenerator.getInstance("DSA");**
- **KeyPair keyPair = keyPairGenerator.generateKeyPair();**

Spring Security

Безопасность веб-приложений

Основные классы

- **SecurityContextHolder** – содержит в себе объект контекст безопасности, который соответствует текущему пользователю(доверителю, Principal), обычно связан с текущим потоком, но может изменять привязку в соответствии со стратегией
- **SecurityContext** – контекст безопасности, который содержит в себе объект аутентификации
- **Authentication** – содержит в себе токен аутентификации, может представлять множество типов аутентификации(пароль, JAAS, OpenID и т.д.)
- **GrantedAuthority** – предоставленное право на действие, которое предоставлено для объекта аутентификации
- **UserDetails** – содержит более менее детальное описание пользователя, ассоциированного с текущим контекстом безопасности(InetOrgPerson, LdapUserDetailsImpl, Person, User)
- **UserDetailsService** – сервис получения информации о пользователе(CachingUserDetailsService, InMemoryUserDetailsManager, JdbcDaoImpl, JdbcUserDetailsManager, LdapUserDetailsManager, LdapUserDetailsService)

JSON Web Token

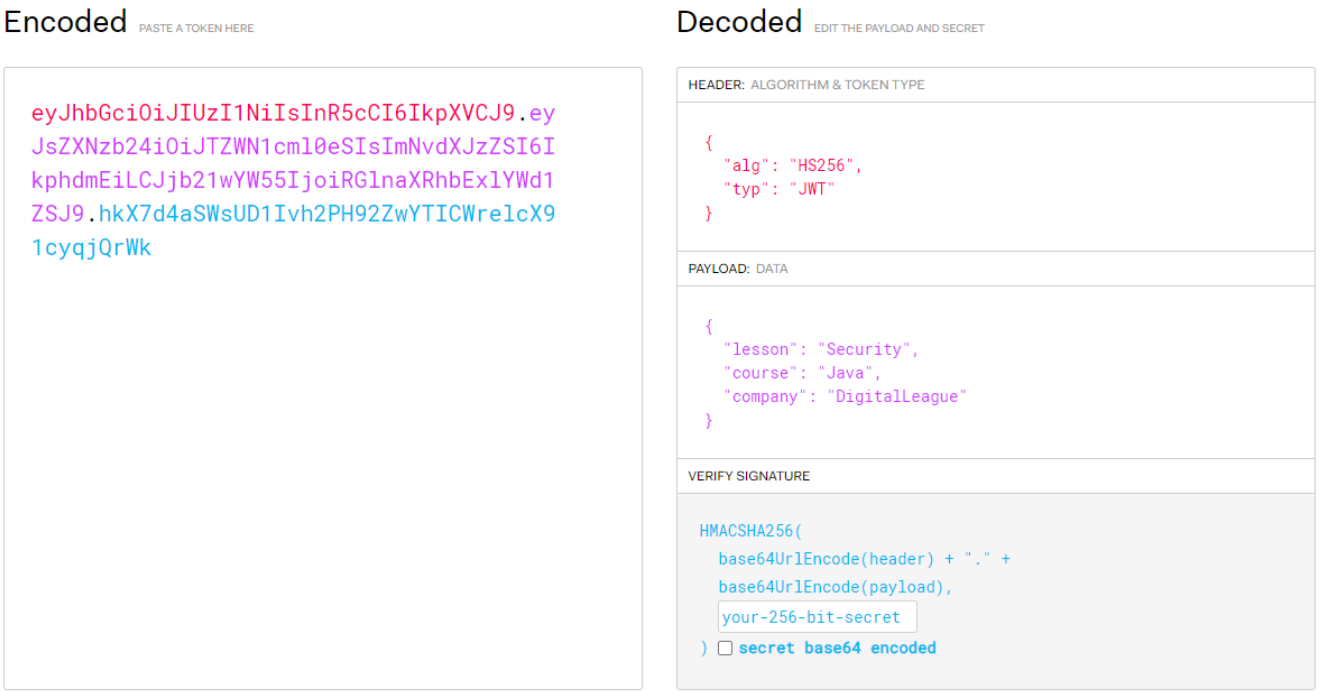
Стандарт

АКАДЕМИЯ

JWT

Описание и структура

- **JSON Web Token** (**JWT**) — открытый стандарт использования токенов доступа в среде клиент-серверных приложений, основанный на формате JSON. Используется для подтверждения аутентификации. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для подтверждения прохождения аутентификации.

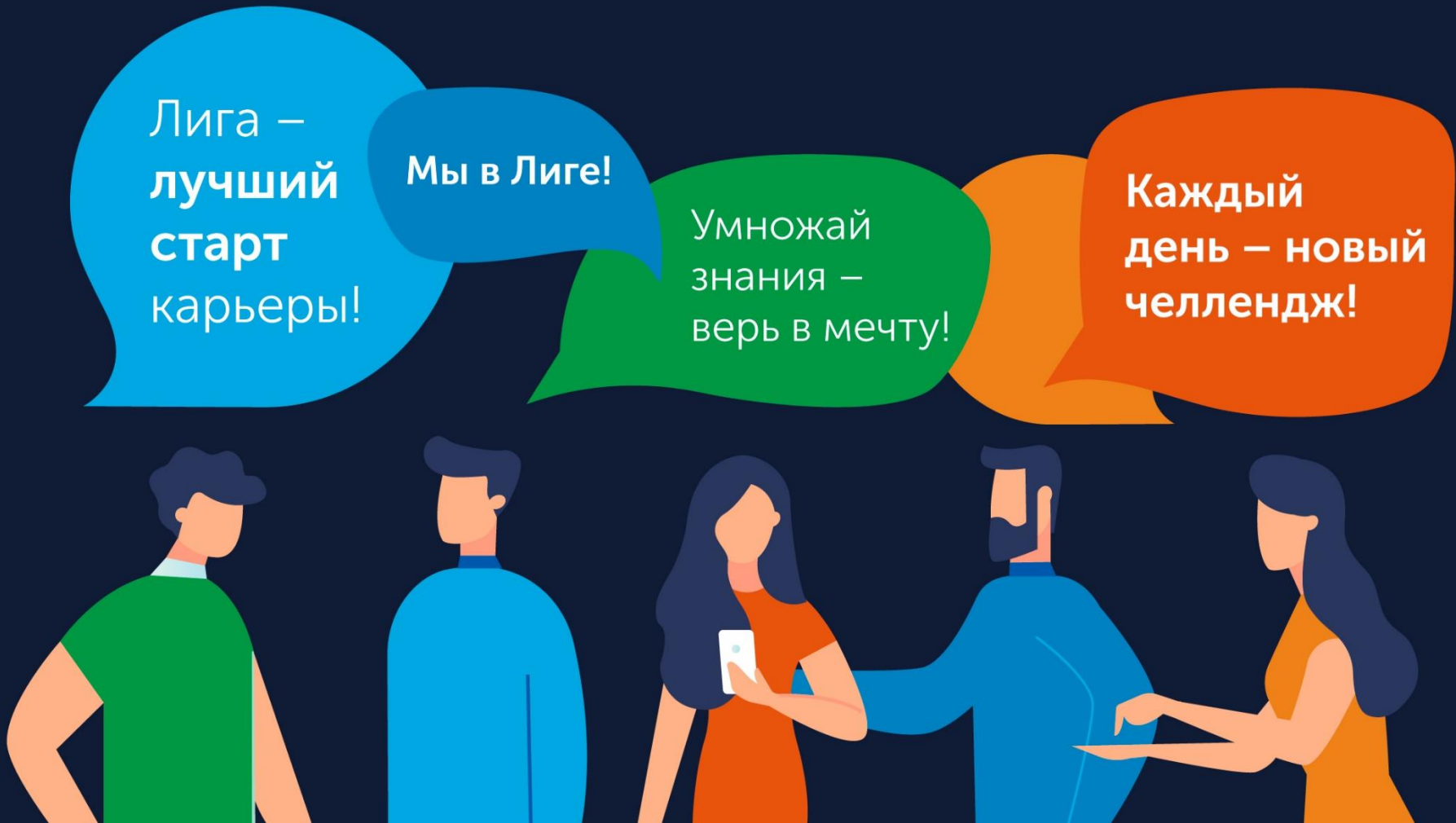


АКАДЕМИЯ

Основные классы

- **Jwts** – класс со статическими утилитарными методами, необходимыми для работы с JWT
- **Jwt** – представление JWT в виде java-объкта
- **JwtBuilder** –предоставляет методы по построению java-объекта JWT, включает в себя формирование заголовка, основного тела и подписывания токена
- **JwtParser** – читает строковое представление JWT и превращает его в java-объект
- **Header** – заголовок JWT
- **Claim** – список пар ключ-значение, из которых состоит тело JWT. Определяется разработчиком под конкретные задачи

Всем спасибо!



Андрей Лабазин

Разработчик

АКАДЕМИЯ