- Dodać dependency do ActiveMQ JMS

- Tworzymy klasę konfiguracyjną dla JMS:

@Configuration

@EnableJms // tylko jeżeli nie używamy zewnętrznego brokera JMS

- W Spring Boocie mozemy dodac annotacje @EnableJms nad glowna klasa aplikacji

- PRODUCER : W ciele klasy definiujemy @Bean zwracający nam queue do której będziemy

wysyłać wiadomości

@Bean

public Queue queue(){

return new ActiveMQQueue("nazwa")

}

- PRODUCER : Musimy skonfigurować ConnectionFactory i przekazać mu m.in. adres brokera

czyli servera JMS na który będziemy wysyłać wiadomośći

@Bean

public ActiveMQConnectionFactory activeMQConnectionFactory(){

ActiveMQConnectionFactory factory = new ActiveMQConnectionFactory();

factory.setBrokerURL(brokerUrl);

return factory;

}

- PRODUCER : W kontrolerze gdzie chcemy używać JMS autowireujemy beana JmsTemplate wbudowanego w Spring.

Ten bean automaczynie będzie stawiał nam ConnectionFactory i wykonywał automatyczną

konfigurację wg ustawień domyślnych.

- PRODUCER : Jeżeli chcemy żeby JmsTemplate używał naszej customowej ConnectionFactory musimy

ją zarejestrować \*w pliku konfiguracyjnym\*

@Bean

public JmsTemplate jmsTemplate(){

//tworzy nowy JmsTemplate przekazując mu

//nasz customowy ConnectionFactory

return new JmsTemplate(activeMQConnectionFactory());

}

- PRODUCER : W kontrolerze na zautowireowanym obiekcie jmsTemplate możemy wywołać metodę .convertAndSend

która przyjmuje jako argument to na co mamy wysłać obiekt i co mamy wysłać, np:

//pobierz wiadomość z URI, wyślij na kolejkę queue

@GetMapping("/{message}")

public String publish(@PathVariable String message){

jmsTemplate.convertAndSend(queue, message);

//jmtTemplate jest zoverloadowany tak ze w pierwszym parametrze

//moze przyjac dwolny obiekt dziedziczacy z klasy Destination, np. Queue albo Topic

//moze tez przyjac String bedacy nazwa kolejki lub topicu na ktory chcemy wyslac obiekt

return "Message published";

}

- Jeżeli podamy w jmsTemplate nazwę destination, która nie istnieje, to JmsTemplate automatycznie ją stworzy

- W klasie, która ma nasłuchiwać definiujemy metodę z annotacją:

@JmsListener(destination = "nazwaKolejki")

Która wykona się w momencie otrzymania obiektu w kolejce

== ROBIENIE TEGO IN MEMORY ==

- W application.properties dodajemy:

spring.activemq.in-memory = true

spring.activemq.pool.enabled = false

server.port = 8081

== ROBIENIE TEGO NA ZEWNĘTRZNYCH SERVERZE ==

- spring.activemq.in-memory = false

spring.activemq.pool.enabled = false

server.port = 8081

activemq.broker-url=tcp://localhost:61616 //domyślny adres ActiveMQ

- ActiveMQ ściąga i odpla się ze strony apache

- Wejście do panelu admina:

http://localhost:8161/admin/