public MessageConverter stringToPersonMessageConverter() {

return new MessageConverter() {

@Override

public Object fromMessage(Message<?> message, Class<?> targetClass) {

Person person = new Person();

person.setName((String) message.getPayload());

return person;

}

@Override

public Message<?> toMessage(Object payload, MessageHeaders headers) {

return MessageBuilder.withPayload(payload).copyHeadersIfAbsent(headers).build();

}

};

}

@Bean

public GenericConverter stringToPersonConverter() {

return new GenericConverter() {

@Override

public Set<ConvertiblePair> getConvertibleTypes() {

return Collections.singleton(new ConvertiblePair(String.class, Person.class));

}

@Override

public Object convert(Object source, TypeDescriptor sourceType, TypeDescriptor targetType) {

Person person = new Person();

person.setName((String) source);

return person;

}

};

}

public static class Person {

private String name;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

return "name: " + name;

}

}