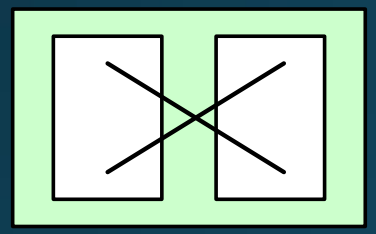
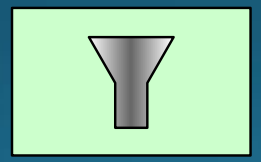


Transformers

Express Spring Integration

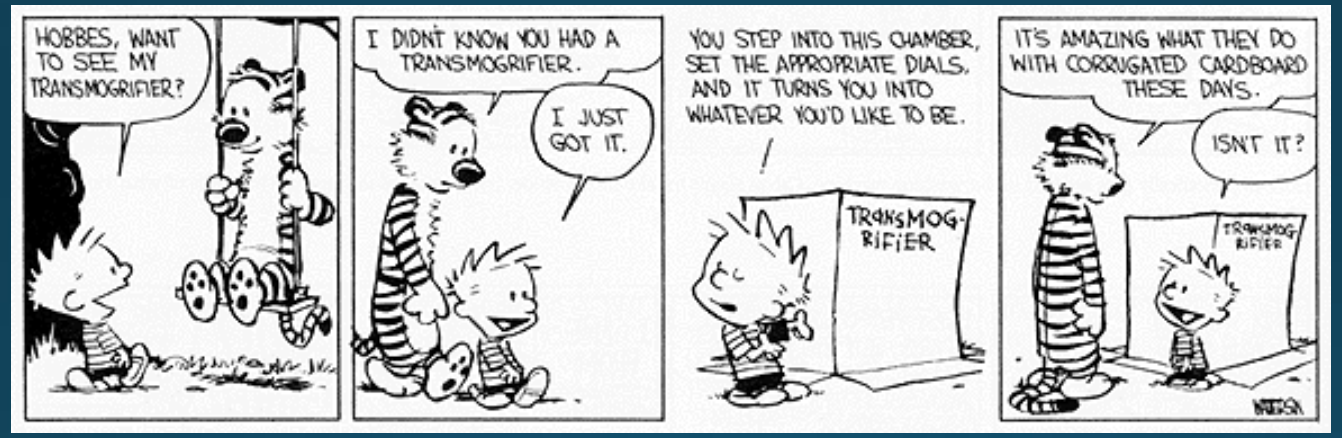
SI Filters Review

- We are in the middle of examining a number of SI message endpoints.
- Filters are endpoints that sit between channels that select or reject messages from one message channel to the next.
  - Messages not selected are discarded.
  - Selection occurs on the basis of message payload or message metadata (header information).
- SI provides many filters out of the box.
- Custom filters can be created with the help of a MessageSelector implementation.

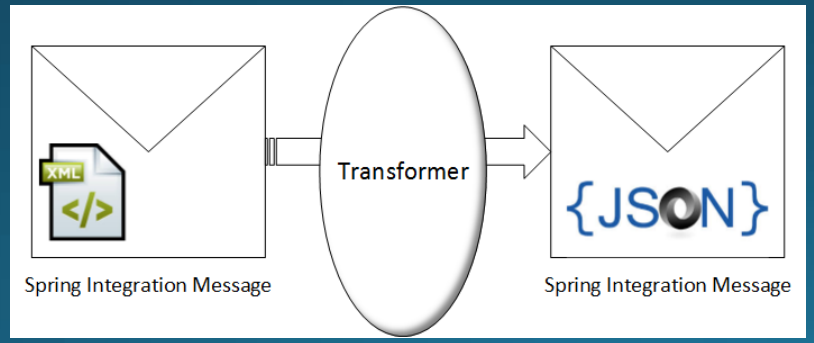


Transformers

- Transformers convert the payload or structure of a message into a modified message.

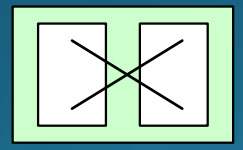


- For example, convert an XML payload message into a JSON payload message.



SI Transformers

- Spring Integration comes with a number of built-in transformers.
  - XML to object (unmarshallers) / object to XML (marshallers)
  - Object to string/string to object
  - file to string/string to file (as seen in the last lab)
  - Object serializer/deserializer
  - Object to map/map to object
  - Object to JSON/JSON to object
  - Claim check (implementing the claim check design pattern)
- Use simple POJOs to create your own custom transformer
- Translators, as they are called by Hophe/Woolf, are represented by icon show here in EIP diagrams.



Simple SI transformers

- An object-to-string transformer

```
<int:object-to-string-transformer input-channel="inboundChannel"  output-channel="outboundChannel‟ />
```

  - Takes a message with an object payload from the inboundChannel
  - Calls the toString( ) method of the payload object
  - Puts a message containing the resulting string in the outboundChannel.
- String to string transformer using SpEL

```
<int:transformer input-channel="inboundChannel"
      output-channel="outboundChannel" expression="payload.toUpperCase()"/>
```


You are ready to tackle Lab 4

- In Lab 4, you work with a couple of simple string transformers. The first will use a built-in transformer with some SpEL to provide the transformation logic. The second has you create your own custom transformer.
- In the third transformer you work with, you see the built-in XML to object unmarshalling transformer, which takes advantage of JAXB under the covers.
- In this lab you will also see the use of a service activator to display the contents of a message. Service activators are another message endpoint type you explore later in this tutorial.
- Finally, as Spring and all of Java has embraced annotations in place of XML configuration, you will see the use of Spring annotations for configuring your SI components.

Associated Courses and Resources

Upcoming Training

- Spring Training
- To learn more :http://bit.ly/1hyrViM

- Click here for associated labs and video