SYMPHONY

Developer Newsletter

This newsletter covers best practices in message templating, the new Symphony Bot Developer Kit (BDK), the next editions of the Innovate Hackathon series, the Symphony Developer Certification, and more.

Develop: Message Templating

Have you written a simple bot that got increasingly complex over time? If so, you might have encountered challenges with constructing message content and maintaining its code. You can construct MessageML in string variables that's embedded within business logic. However, this quickly becomes unsustainable as the project grows. As a best practice, Symphony recommends using message templating to ensure message structures are more easily maintained in the future. This applies for smaller projects as well.

The core benefit of templating is to isolate static message content from business logic. Templating also allows for some useful flow control syntax, such as loops, to facilitate data-driven content. This reduces the amount of boilerplate and string duplication within your code. There are a number of templating engines that range in popularity, depending on your preferred language or framework. We will discuss two examples in the context of building Symphony bots in Java: **Apache FreeMarker** and **Handlebars**.

Apache FreeMarker is built into Symphony, so it doesn't require engine-specific dependencies. You can use `sendMessage()` in the SDKs or the corresponding create message REST API call to send FreeMarker syntax directly in the message field with the string representation of your object in the data field.

SYMPHONY

Developer Newsletter

FreeMarker template saved in resources/template.ftl

```
<div>
  Status: ${data.status} of ${data.items?size} items
  <#list data.items as item>
    <div><b>Item ${item?index+1}</b>: ${item.name} x ${item.quantity}</div>
  </#list>
</div>
```

JSON representation of data object

```
{
  "status": "Pending Delivery",
  "items": [
    { "name": "Maki", "quantity": 150 },
    { "name": "Sake", "quantity": 200 },
  ]
}
```

Java Code for loading a FreeMarker template and sending it in a message

```
// Load template
Path path = Paths.get(getClass().getResource("/template.ftl").toURI());
String template = String.join("\n", Files.readAllLines(path));

// Build data object and stringify using jackson-core
List<OrderItems> items = Arrays.asList(
  new OrderItem("Maki", 150),
  new OrderItem("Sake", 200)
);
Order order = new Order("Pending Delivery", items);
String data = (new ObjectMapper()).writeValueAsString(order);

// Send message
OutboundMessage outboundMessage =  new OutboundMessage(template, data);
botClient.getMessagesClient().sendMessage(streamId, outboundMessage);
```

The MessageML string is reduced significantly as the number of data objects grow, owing to the `<#list/>` construct. There are also other constructs like `if/else` or `?size` that enables the template to dynamically adjust to the data content without any string manipulation.

Unlike FreeMarker, Handlebars is independent of Symphony, so it requires an additional dependency in your project. It also requires an extra compilation step to build the actual MessageML string which will be used in `sendMessage()` without the data field.

Java Code for compiling a Handlebars template and sending it in a message

```
// Compile message
Handlebars handlebars = new Handlebars();
String message = handlebars.compileInline(template).apply(order);

// Send message
OutboundMessage outboundMessage =  new OutboundMessage(message);
botClient.getMessagesClient().sendMessage(streamId, outboundMessage);
```

Handlebars also differs in that it is based on a principle of logicless templates. This means that less flow control syntax is available, but it allows you to define custom helpers in your code. For example, if you'd like to add 1 to the 0-based `@index` in your template, you can define a `plusOne` helper as follows:

Handlebars template

```
{{#each items}}
  Item number {{plusOne @index}} is {{name}}
{{/each}}
```

Java Code for helper registration
`handlebars.registerHelper("plusOne", (value, options) -> value + 1);`

Each engine has its strengths, so choose one that fits your needs. You can find a more detailed and complete guide on templating in Java at our **developer documentation** site. The page also includes a complete sample project for you to try out.

Symphony Bot Developer Kit (BDK)

Symphony is pleased to provide our developer community a sneak preview of the soon-to-be-released **Bot Developer Kit (BDK)**.

Some developers in our community are familiar with the Symphony Generator, which provides sample code, a project scaffold and a customized config.json file. The Symphony BDK goes further by providing developers with a collection of tools to help quickly build and deploy production-ready bots and applications.

As Symphony develops and deploys fully interactive integrations, our development team will continue to compile best practices and tools into the BDK. The underlying Bot SDK is heavily Java/Spring-based, while the UI ToolKit is React-based.

The Bot Developer Kit contains the following tools:

- **CLI** - to easily create new projects, and quickly add commands or message templates to an existing bot project

- **Bot SDK** - including the following features to:
    - handle bot commands
    - connect with external systems (includes handling authentication)
    - send templated messages
    - create advanced command matching patterns
    - create and manage webhooks for notifications from external systems
    - deploy Extension Applications (app.js, controller.js, and configurations),
    - enforce rate limiting
    - and many more
- **UI ToolKit** - a library of visual components (with the sample code for each component!) enabling rich, interactive and consistent Extension Applications

The Bot SDK includes a **CommandHandler** class that allows developers to easily define matching patterns for bot commands and bot response behavior. This allows you to focus on building out business logic, instead of writing parsers for incoming Symphony messages and constructing responses, thus speeding up the bot development process.

The **CommandHandler** has three subclasses (for now):

- **DefaultCommandHandler** to provide default friendly responses
- **AuthenticatedCommandHandler** to manage interactions with external, authenticated systems
- **MultiResponseCommandHandler** to allow bots to respond to commands posted in one room and broadcast them to multiple rooms

SYMPHONY

Developer Newsletter

Register: Symphony Innovate Europe Hackathon 2020 in Paris

Build on our secure collaboration platform to extend critical workflows beyond your company walls. Develop scalable, innovative solutions at an upcoming Symphony Hackathon to bring efficiency to your firm. Such solutions should break down silos across firms or internally across front, middle, and back offices. Gather your team of 1-5 individuals and **register** for the Paris Hackathon at **BPIfrance** on **July 1, 2020**.

Coding will begin at 9am and end at 5pm, followed by presentations, an awards ceremony, and cocktail hour.

Winning teams and their solutions will be featured at **Symphony Innovate New York on 7-8 October.**

For more information or to view previous Hackathon Award winners, please visit our **Innovate Hackathon website**.

Announcing the Symphony Developer Certification

Are you ready to start your Symphony Developer Certification Journey? In April 2020, we will introduce the Symphony Developer Certification program, along with a suite of educational course content.

Follow the course materials to accelerate your learning experience while progressing through the course levels:

- **Beginner**: Symphony Development Fundamentals
- **Intermediate**: Developing Bots and Apps with Symphony
- **Advanced**: Symphony Advanced Development

Look out for further announcements on this exciting new material coming out in **April 2020**!

Developer Events

Make sure to join one of our **Symphony Developer Meetup Groups** to receive updates on upcoming developer events near you!

Share with a Colleague

Know a colleague that will find the developer newsletter useful? Help them **subscribe to the newsletter** now.