GAE

Camel Components for Google App Engine

This component is deprecated and will be removed form Camel 2.18 onwards.

Tutorials

- A good starting point for using Camel on GAE is the Tutorial for Camel on Google App Engine
- The OAuth tutorial demonstrates how to implement OAuth in web applications.

The Camel components for Google App Engine (GAE) are part of the `camel-gae` project and provide connectivity to GAE's cloud computing services. They make the GAE cloud computing environment accessible to applications via Camel interfaces. Following this pattern for other cloud computing environments could make it easier to port Camel applications from one cloud computing provider to another. The following table lists the cloud computing services provided by Google and the supporting Camel components. The documentation of each component can be found by following the link in the *Camel Component* column.

GAE service	Camel component	Component description
URL fetch service	ghttp	Provides connectivity to the GAE URL fetch service but can also be used to receive messages from servlets.
Task queueing service	gtask	Supports asynchronous message processing on GAE by using the task queueing service as message queue.
Mail service	gmail	Supports sending of emails via the GAE mail service. Receiving mails is not supported yet but will be added later.
Memcache service		Not supported yet.
XMPP service		Not supported yet.
Images service		Not supported yet.
Datastore service		Not supported yet.
Accounts service	gauth glogin	These components interact with the Google Accounts API for authentication and authorization. Google Accounts is not specific to Google App Engine but is often used by GAE applications for implementing security. The gauth component is used by web applications to implement a Google-specific OAuth consumer. This component can also be used to OAuth-enable non-GAE web applications. The glogin component is used by Java clients (outside GAE) for programmatic login to GAE applications. For instructions how to protect GAE applications against unauthorized access refer to the Security for Camel GAE applications page.

Camel context

Setting up a `SpringCamelContext` on Google App Engine differs between Camel 2.1 and higher versions. The problem is that usage of the Camel-specific Spring configuration XML schema from the `http://camel.apache.org/schema/spring` namespace requires JAXB and Camel 2.1 depends on a Google App Engine SDK version that doesn't support JAXB yet. This limitation has been removed since Camel 2.2.

JMX must be disabled in any case because the `javax.management` package isn't on the App Engine JRE whitelist.

Camel 2.1

`camel-gae` 2.1 comes with the following `CamelContext` implementations.

- `org.apache.camel.component.gae.context.GaeDefaultCamelContext` (extends `org.apache.camel.impl.DefaultCamelContext`)
- `org.apache.camel.component.gae.context.GaeSpringCamelContext` (extends `org.apache.camel.spring.SpringCamelContext`)

Both disable JMX before startup. The `GaeSpringCamelContext` additionally provides setter methods adding route builders as shown in the next example.

appctx.xml

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd">

    <bean id="camelContext"
        class="org.apache.camel.component.gae.context.GaeSpringCamelContext">
        <property name="routeBuilder" ref="myRouteBuilder" />
    </bean>

    <bean id="myRouteBuilder"
        class="org.example.MyRouteBuilder">
    </bean>

</beans>
```

Alternatively, use the `routeBuilders` property of the `GaeSpringCamelContext` for setting a list of route builders. Using this approach, a `SpringCamelContext` can be configured on GAE without the need for JAXB.

Camel 2.2 or higher

With Camel 2.2 or higher, applications can use the `http://camel.apache.org/schema/spring` namespace for configuring a `SpringCamelContext` but still need to disable JMX. Here's an example.

appctx.xml

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:camel="http://camel.apache.org/schema/spring"
       xsi:schemaLocation="
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
http://camel.apache.org/schema/spring
http://camel.apache.org/schema/spring/camel-spring.xsd">

    <camel:camelContext id="camelContext">
        <camel:jmxAgent id="agent" disabled="true" />
        <camel:routeBuilder ref="myRouteBuilder"/>
    </camel:camelContext>

    <bean id="myRouteBuilder"
        class="org.example.MyRouteBuilder">
    </bean>

</beans>
```

The web.xml

Running Camel on GAE requires usage of the `CamelHttpTransportServlet` from `camel-servlet`. The following example shows how to configure this servlet together with a Spring application context XML file.

web.xml

```
<web-app
xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
xsi:schemaLocation="
http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" version="2.5">

    <servlet>
        <servlet-name>CamelServlet</servlet-name>
        <servlet-class>org.apache.camel.component.servlet.CamelHttpTransportServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>appctx.xml</param-value>
        </init-param>
    </servlet>

    <!--
        Mapping used for external requests
     -->
    <servlet-mapping>
        <servlet-name>CamelServlet</servlet-name>
        <url-pattern>/camel/*</url-pattern>
    </servlet-mapping>

    <!--
        Mapping used for web hooks accessed by task queueing service.
     -->
    <servlet-mapping>
        <servlet-name>CamelServlet</servlet-name>
        <url-pattern>/worker/*</url-pattern>
    </servlet-mapping>

</web-app>
```

The location of the Spring application context XML file is given by the `contextConfigLocation` init parameter. The `appctx.xml` file must be on the classpath. The servlet mapping makes the Camel application accessible under `http://<appname>.appspot.com/camel/...` when deployed to Google App Engine where `<appname>` must be replaced by a real GAE application name. The second servlet mapping is used internally by the task queueing service for background processing via web hooks. This mapping is relevant for the gtask component and is explained there in more detail.