AWS Developer Tools Blog

Introducing Smithy for Python

by Jordon Phillips | on 25 JUL 2023 | in Announcements, Developer Tools, Python | Permalink | Comments | Share

AWS is excited to announce a preview of Smithy client generation for Python. This tooling will enable developers to generate clients in type-hinted Python in the same model-driven manner that AWS has used to develop its services for more than a decade. Writing and maintaining hand-written clients for a web service is both time-consuming and error-prone, problems which are both solved by generating clients with an unambiguous specification. We have made Smithy for Python open-source to allow any developer to confidently and quickly generate their own clients.

What is Smithy?

Smithy is an open-source toolchain that we've standardized on to build new AWS SDKs, model new AWS services and features, and to generate clients. Smithy includes a protocol-agnostic Interface Definition Language (IDL) for generating clients, servers, documentation, and other artifacts. To learn more, check out smithy.io, and please watch the introductory talk from Michael Dowling, Smithy's Principal Engineer.

What's included?

A code generated Python client will have the following features:

- Code generated and type hinted – Code for clients, operations, and all the data they contain will be code generated into python classes. Along with type hints, this will allow customers to easily lint and debug their code using tools like mypy or other type analysis tooling.
- Interceptors – Interceptors are the new extensibility feature that replace the botocore event system. They're designed to be more reliable and to be consistent across all of Smithy's supported languages.
- Async first – Async is increasingly important in the Python ecosystem, and since clients spend much of their time blocked on IO they're the perfect use case for it. All operations will therefore be generated as async functions.
- Configurable components – All of the core components of the client will be fully configurable. If you want to use tornado as the underlying http library, for example, you can easily do that.

Currently protocol support is limited to the restJson1 protocol, which makes use of http bindings with a JSON body. It's widely used among Amazon's services, such as API Gateway and Lambda. For authentication, httpApiKey auth is currently supported.

Getting started

There are three prerequisites to client generation. The first prerequisite is Python, which has a minimum required version of 3.11. The second prerequisite is Java 17, which is only required to build the generator. In the future, when the generator is published to a package repository, this will not be required. It is also not required to run the generated Python client. The last prerequisite you'll need is the Smithy CLI.

To build the generator, clone the smithy-python GitHub repo and run `make install-java-components`. This will build the generator and make it available on your local machine.

Now that you've built the generator, you need a Smithy model. In this example, you'll use a small model for a service that accepts a string message in a JSON body and returns it unmodified. Save the following model contents to a file called `main.smithy` in a directory called `model`.

Smithy

```
$version: "2.0"

namespace com.example

use aws.protocols#restJson1

@restJson1
service EchoService {
    operations: [Echo]
}

@http(
    method: "POST"
    uri: "/echo"
)
operation EchoMessage {
    input := {
        message: String
    }
```

Now you need to create a configuration file named `smithy-build.json`. This configuration file lets you specify dependencies, configure plugins, and create projections, which are different views of your model catered to a specific purpose. Additional information on what you can do with this configuration file and projections is provided in the smithy-build docs. The following `smithy-build.json` file adds a few necessary dependencies and creates a projection to build the python client in. Create your `smithy-build.json` file in the same directory as your model folder with the following contents:

JSON

```
{
    "version": "1.0",
    "sources": ["model"],
    "maven": {
        "dependencies": [
            "software.amazon.smithy:smithy-model:[1.34.0,2.0)",
            "software.amazon.smithy:smithy-aws-traits:[1.34.0,2.0)",
            "software.amazon.smithy.python:smithy-python-codegen:0.1.0"
        ]
    },
    "projections": {
        "python-client": {
            "plugins": {
                "python-client-codegen": {
                    "service": "com.example#EchoService",
                    "module": "echo",
                    "moduleVersion": "0.1.0"
                }
            }
```

Your file structure now should now look like this:

```
.
├── model
│   └── main.smithy
└── smithy-build.json
```

The only step left to generate your client is to run `smithy build`. Your new client will now be available as an installable package in `build/python-client/python-client-codegen`. To use it, you'll need to install a few Python dependencies by running `make install-python-components` from the root of the smithy-python repository. The following snippet shows how you can use your new client:

Python

```python
import asyncio

from echo.client import EchoService
from echo.config import Config
from echo.models import EchoMessageInput


async def main() -> None:
    client = EchoService(Config(endpoint_uri="https://example.com/"))
    response = await client.echo(EchoMessageInput(message="spam"))
    print(response.message)


if __name__ == "__main__":
    asyncio.run(main())
```

What's next for Smithy for Python?

We will be delivering new features to the code generator to make generating custom clients easier. We will also be delivering support for more protocols and features to enable generation for more kinds of services.

Feedback

We encourage you to try out Smithy for Python to build your own clients. If you have any question, comments, concerns, or ideas, please open an issue or bug report on the smithy-python GitHub repository. If you discover a potential security issue, we ask that you notify AWS/Amazon Security via our vulnerability reporting page instead. We appreciate any feedback we receive, and will use it to make the next versions of the clients better.

TAGS: Python, SDK, smithy

Comments