

Generating SDKs from OpenAPI

Lorna Mitchell, Vonage



What is OpenAPI?

Machine-readable API description (YAML or JSON)

- Describe endpoints, with verbs
- Parameters, request bodies
- Responses

The standard formerly known as Swagger



Why Describe Your APIs?

- Publish API descriptions for developers to use
- Generate API reference docs
- Import to Postman
- Use with mock servers

- Generate SDKs and libraries

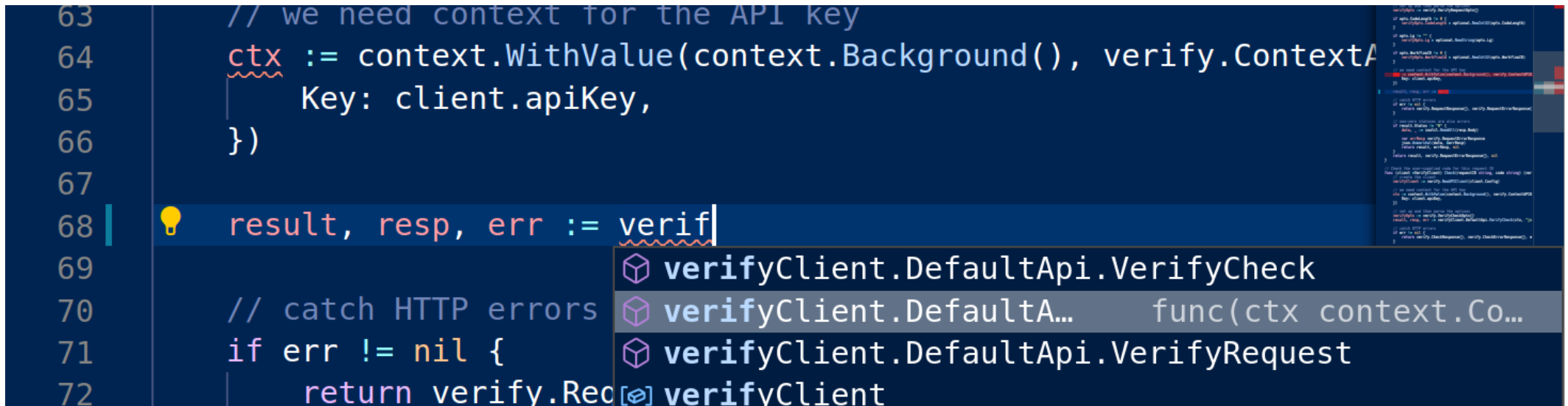


SDKs for Delightful Developer Experiences

Don't Make Me Think

Reduce cognitive load on your users

```
63 // we need context for the API key
64 ctx := context.WithValue(context.Background(), verify.ContextA
65     Key: client.apiKey,
66 })
67
68 ⚡ result, resp, err := verify
69     verifyClient.DefaultApi.VerifyCheck
70 // catch HTTP errors
71 if err != nil {
72     return verify.Reco[verifyClient
```



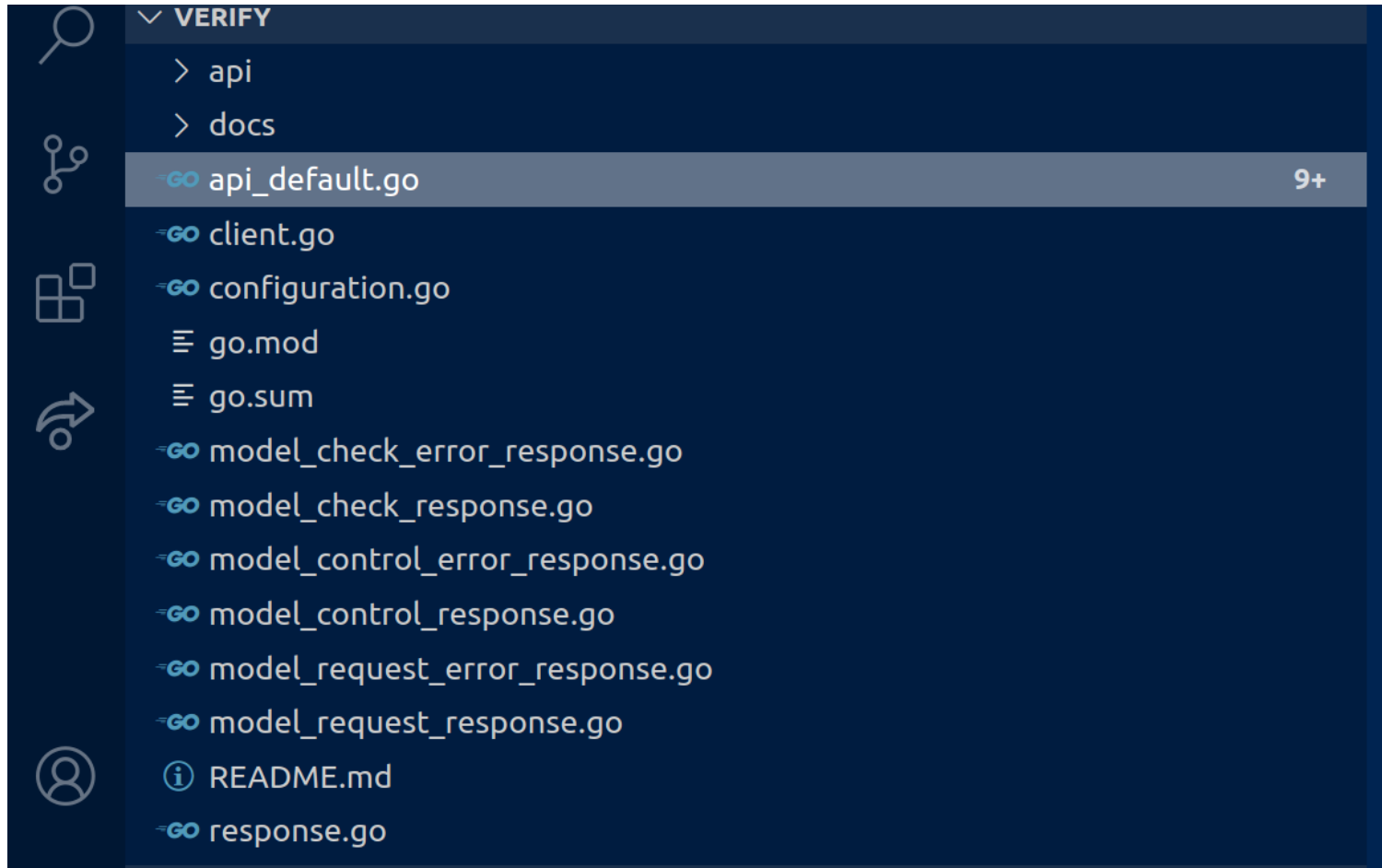
How to Generate Code

This example from <https://openapi-generator.tech/>

```
docker run --rm -v ${PWD}:/local openapitools/openapi-generator-cli \  
generate \  
-i verify.yml \  
--package-name verify \  
-g go \  
-o /local/out/golang/verify
```



Generated Code



A screenshot of a file explorer interface showing a project structure. The root directory is 'VERIFY', which is expanded to show subdirectories 'api' and 'docs'. Below these are several Go source files, each with a small Go logo icon to its left. The file 'api_default.go' is highlighted in a light blue bar and has a '9+' badge to its right. Other files include 'client.go', 'configuration.go', 'go.mod', 'go.sum', 'model_check_error_response.go', 'model_check_response.go', 'model_control_error_response.go', 'model_control_response.go', 'model_request_error_response.go', 'model_request_response.go', 'README.md', and 'response.go'. On the left side of the explorer, there are icons for search, branching, grid view, refresh, and user profile.

```
✓ VERIFY
  > api
  > docs
  -GO api_default.go 9+
  -GO client.go
  -GO configuration.go
  ≡ go.mod
  ≡ go.sum
  -GO model_check_error_response.go
  -GO model_check_response.go
  -GO model_control_error_response.go
  -GO model_control_response.go
  -GO model_request_error_response.go
  -GO model_request_response.go
  ⓘ README.md
  -GO response.go
```

Generated Code

Generated libraries are better than nothing!

If there is no library, a code generator gives a developer:

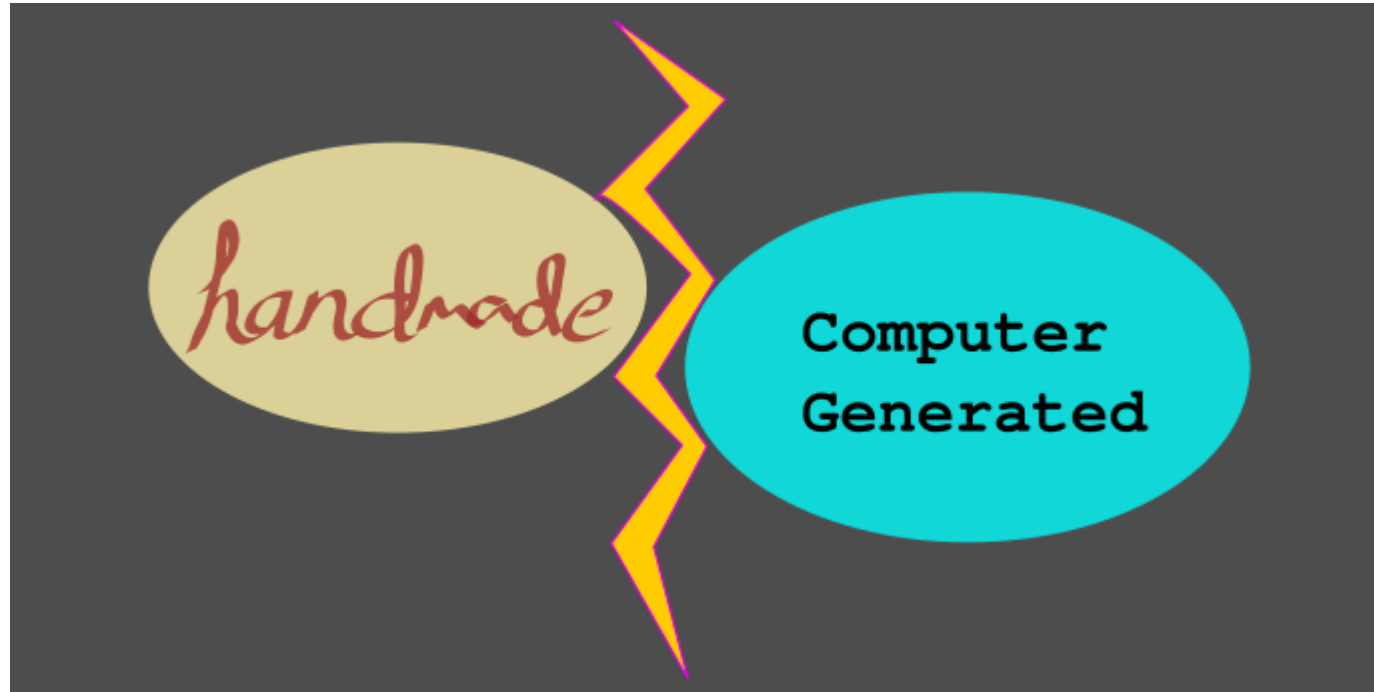
- Abstracts away the HTTP handling
- Simple validation (probably)
- IDE autocomplete for the basic structure



We Can Do Better



About Generated Code



Wrap Sharp Edges

For example, the autocomplete example from earlier:

```
ctx := context.WithValue(  
    context.Background(),  
    verify.ContextAPIKey, verify.APIKey{Key: client.apiKey}  
)  
  
result, resp, err := verifyClient.DefaultApi.VerifyRequest(  
    ctx, "json", client.apiSecret,  
    number, brand, verify.VerifyRequestOpts{ })
```



Wrap Sharp Edges

For the user, it's more like:

```
auth := vonage.CreateAuthFromKeySecret(API_KEY, API_SECRET)
verifyClient := vonage.NewVerifyClient(auth)

resp, _, _ := verifyClient.Request(
    PHONE, "GoTest", vonage.VerifyOpts{ })
```

Auth Helpers

Supply auth details:

```
// for basic key/secret  
auth := vonage.CreateAuthFromKeySecret("abcd1234", "VeryS3cr3t")  
  
// for JWT auth  
privateKey, _ := ioutil.ReadFile(PATH_TO_PRIVATE_KEY_FILE)  
auth, _ := vonage.CreateAuthFromAppPrivateKey(APP_ID, privateKey)
```

Quickly get a JWT:

```
privateKey, _ := ioutil.ReadFile("private.key")  
g := jwt.NewGenerator(APPLICATION_ID, privateKey)  
token, _ := g.GenerateToken()
```

Security Helpers

SMS API webhooks use signatures from shared secrets

```
if smsClient.checkSig(params, SIG_SECRET, "sha256") {  
    // data is plausible, do something cool  
}
```

Make the easy thing also the Right Thing (TM)

Builders

For example, formatting JSON structures from strongly-typed languages

```
MyNcco := ncco.Ncco{}
talk := ncco.TalkAction{
    Text: "Go library calling to say hello",
    VoiceName: "Nicole" }
MyNcco.AddAction(talk)

endpoint := []ncco.PhoneEndpoint{Number: "44777000777"}
connect := ncco.ConnectAction{Endpoint: endpoint, From: "44777000888"}
MyNcco.AddAction(connect)

result, _, err := client.CreateCall(
    vonage.CreateCallOpts{From: from, To: to, Ncco: MyNcco})
```

Delightful SDKs

OpenAPI lets us focus on the best bits!



Thanks!

References:

- <https://github.com/Vonage/vonage-go-sdk>
- <http://spec.openapis.org/>
- <https://developer.nexmo.com/tools>
- <https://openapi.tools>

Me:

- <https://lornajane.net>

