



THE DEVELOPER'S CONFERENCE

Trilha – API

Juscélio Reis
Desenvolvedor

Agenda



- Sobre
- Case Wiz Soluções
- Inspiração
- Modelos
 - Contract First
 - Code First
- Erros

Sobre o palestrante



➤ Juscélio Reis

➤ Pai

➤ Geek

➤ +10 anos de
experiencia em
desenvolvimento de
software.

➤ Pesquisador

➤ Mas....

➤ Wiz Soluções



Wiz Soluções

Advisor Back-end (03/2020 – atual)

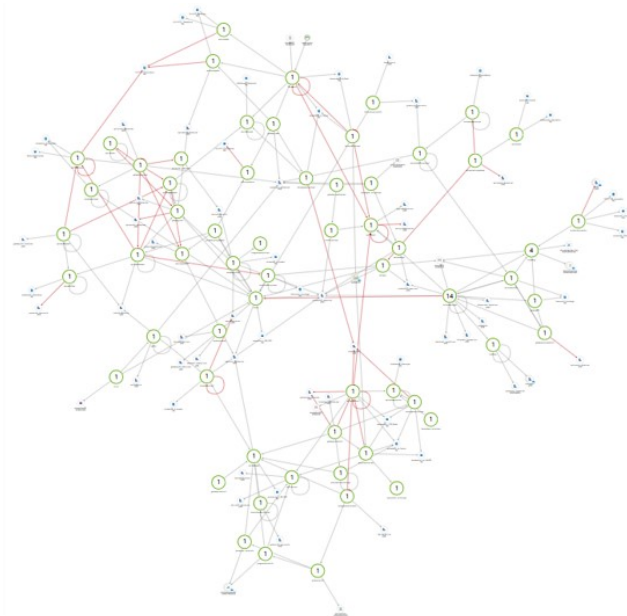
“O nosso Advisor será responsável por desenvolver e implementar o Framework de Back-end e será o guardião da metodologia utilizada em todo o conglomerado Wiz.”

“... Estar antenado no mercado de APIs, frameworks e funcionalidades prontas para trazer as melhores soluções para a Wiz.”

Foco em governança.



THE
DEVELOPER'S
CONFERENCE



WIZ

WIZ
CORPORATE

WIZ
Parceiros

WIZ
BPO

WIZ
Benefícios

WIZ
conseg

WIZ
B2U

Inspirações

Em agosto de 2020 no TDC SP assisti uma palestra sobre governança e uma ferramenta chamada Spectral (Open Source).

E mais muito conteúdo na Web depois.....



Trilha API

Estejam plugados nas novidades das APIs em tempos de transformações contínuas

As transformações se fazem de forma contínua e rápida nas empresas, prover APIs é a melhor maneira de unir Negócios, Tecnologia e Cliente. Vamos discutir na trilha importantes pontos do momento e que estão por vir no Gerenciamento Full Lifecycle de APIs.

Data

📅 Quarta-feira, 26 de Agosto de 2020

🕒 09h às 19h (somente ao vivo)

12:25 às
13:00

Escalando a governança de APIs com automação de processos de validação

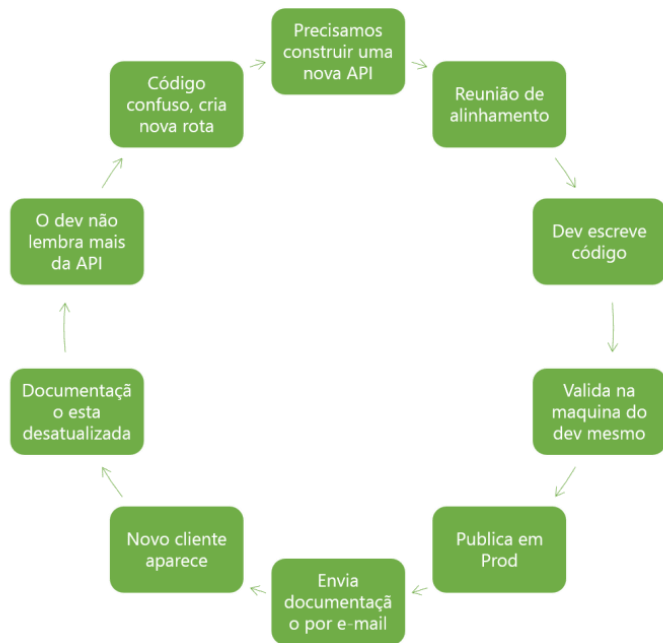
📍 TRILHA

Igor Emmanuel Silva Mendonça / Humberto Corrêa da Silva

A demanda por serviços digitais, torna a integração entre sistemas uma necessidade primária para atuação no mercado. Ferramentas de gestão de APIs são formas de suprir tal necessidade. Porém em cenários de grande escala de APIs, os times de governança, responsáveis por definir as diretrizes do ciclo de vida das APIs, são fortemente exigidos e se tornam gargalos no processo de entrega de valor. Para resolver esse problema, técnicas como Design Guide, aliadas a processos automatizados de validação, como pipelines CI/CD ou linters, podem ser usados para dar suporte e dinamismo a construção de contratos de APIs. Nesta palestra, vamos mostrar, na prática, como esse cenário pode ser implementado.



Code First

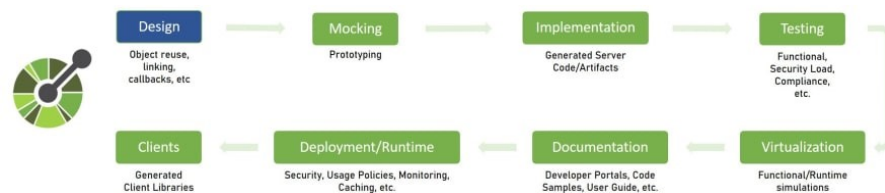


- Tradicional.
- Devs estão acostumado.
- Area comercial não vê problema.
- Problema em atualizar etapas como SDK e doc.

Contract First



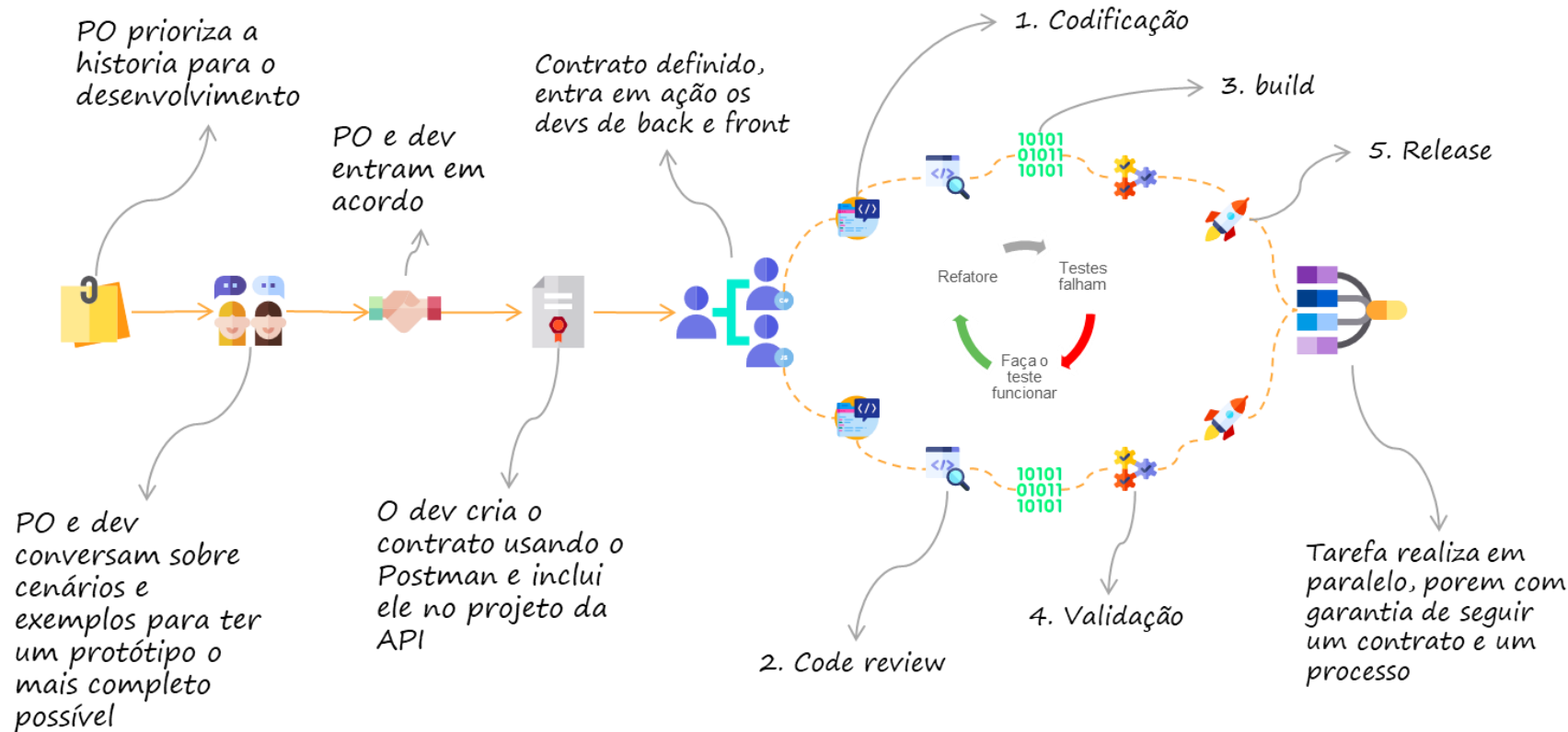
- Excelente oportunidade para colher feedbacks importantes.
- Dev backend e frontend podem trabalhar em paralelo.
- Facilita automação de algumas etapas: sdk, doc, cliente, server.
- Mas pode ser mais demorado. (não acho, mas outras pessoas acham)



Primeira versão



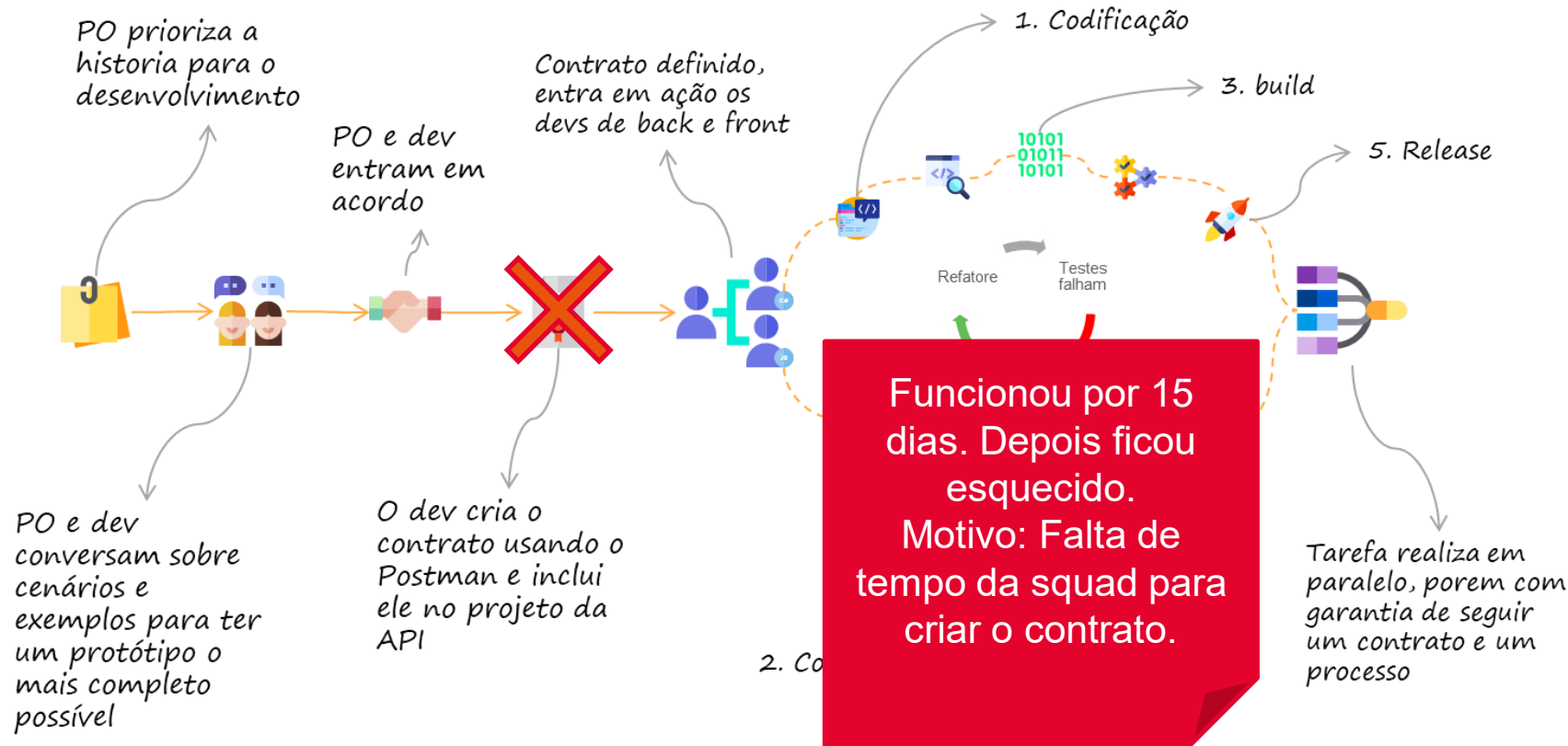
THE
DEVELOPER'S
CONFERENCE



Primeiro fracasso



THE
DEVELOPER'S
CONFERENCE

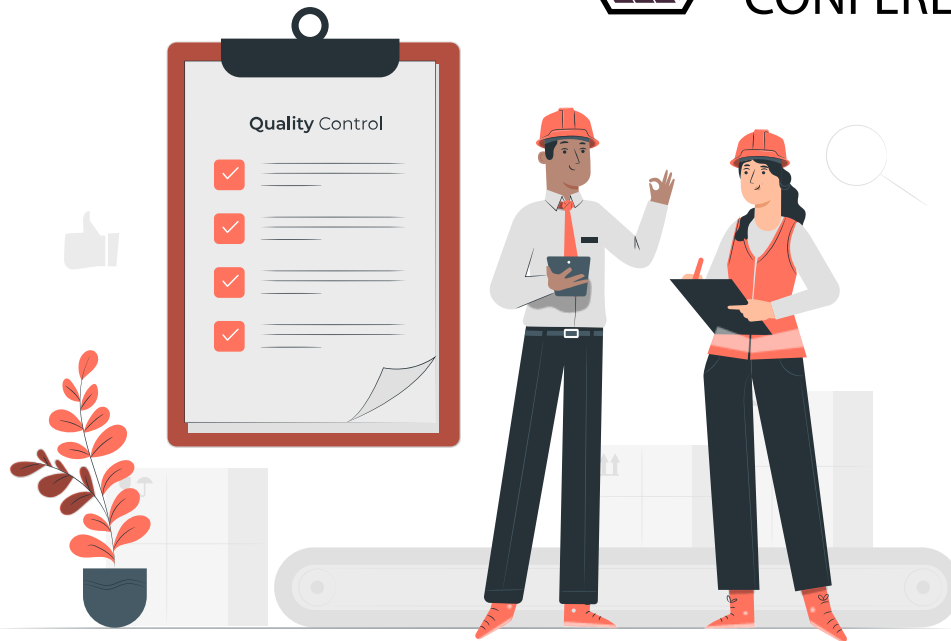




THE
DEVELOPER'S
CONFERENCE

Lições aprendidas

- Não ter autoridade funcional
- Area comercial focada em outras entregas.
- Desenvolvedores fazem quando querem.
- Como garantir que sigam uma metodologia.
- Fast track quebra o fluxo.

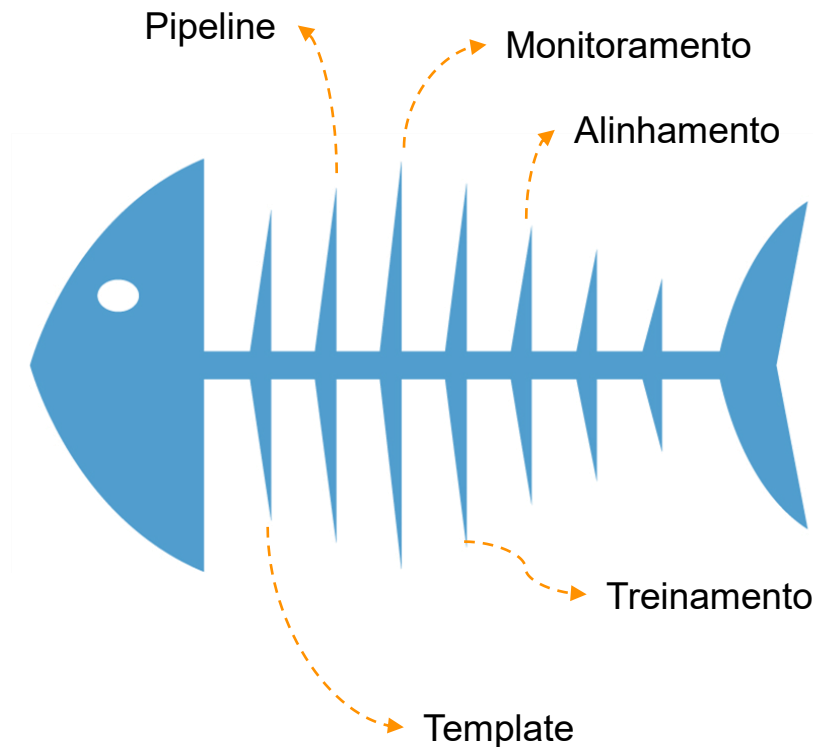




Construir a v2

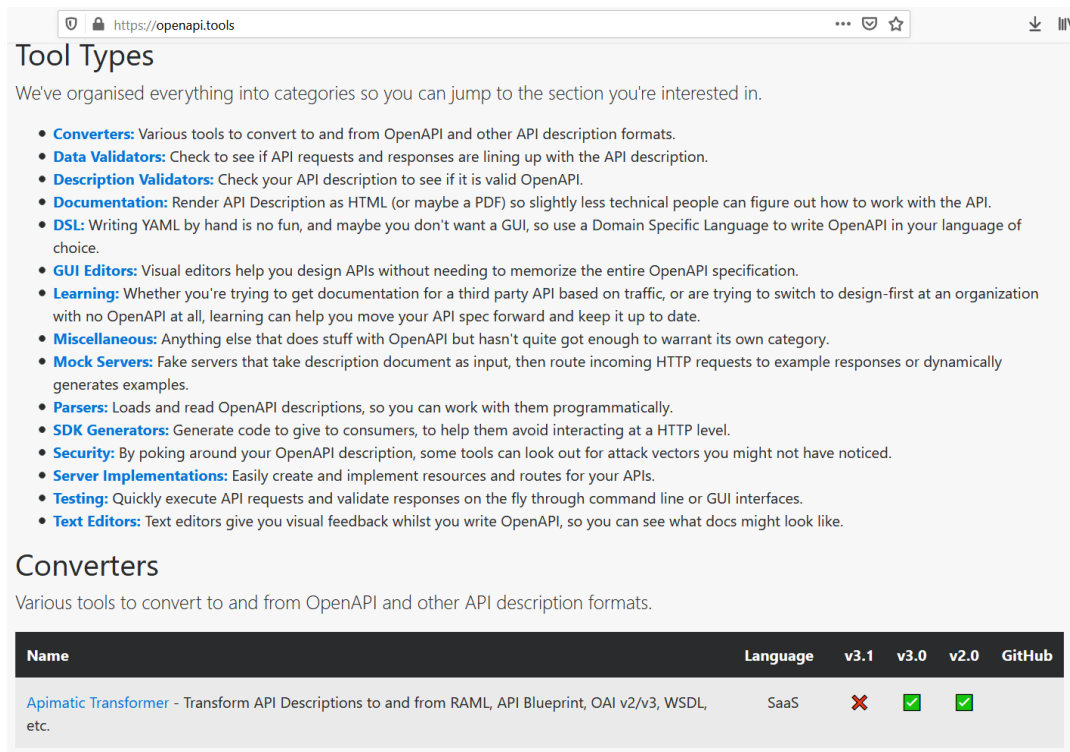
- O processo de CI/CD esta sob meu controle.
- Spectral é uma ferramenta de linha de comando.
- Existem outras ferramentas que geram OpenAPI do código. Ex: Nswag.
- Temos template para criar APIs.

```
> dotnet new -i Wiz.Dotnet.Template.API  
> dotnet new wizapi -n [NomeProjeto]
```



O poder do padrão

- OpenAPI
 - OpenAPI v2 (Swagger)
 - OpenAPI v3
 - OpenAPI v3.1
- API Blueprint
- RAML



The screenshot shows the openapi.tools website. The browser address bar displays 'https://openapi.tools'. The page title is 'Tool Types'. Below the title, a paragraph states: 'We've organised everything into categories so you can jump to the section you're interested in.' A list of tool categories follows, each with a brief description. The categories include: Converters, Data Validators, Description Validators, Documentation, DSL, GUI Editors, Learning, Miscellaneous, Mock Servers, Parsers, SDK Generators, Security, Server Implementations, Testing, and Text Editors. Below the list, the 'Converters' section is expanded, showing a table of tools. The table has columns for Name, Language, v3.1, v3.0, v2.0, and GitHub. The first row shows 'Apimatic Transformer - Transform API Descriptions to and from RAML, API Blueprint, OAI v2/v3, WSDL, etc.' with a language of 'SaaS', a red 'X' for v3.1, and green checkmarks for v3.0 and v2.0.

Tool Types

We've organised everything into categories so you can jump to the section you're interested in.

- **Converters:** Various tools to convert to and from OpenAPI and other API description formats.
- **Data Validators:** Check to see if API requests and responses are lining up with the API description.
- **Description Validators:** Check your API description to see if it is valid OpenAPI.
- **Documentation:** Render API Description as HTML (or maybe a PDF) so slightly less technical people can figure out how to work with the API.
- **DSL:** Writing YAML by hand is no fun, and maybe you don't want a GUI, so use a Domain Specific Language to write OpenAPI in your language of choice.
- **GUI Editors:** Visual editors help you design APIs without needing to memorize the entire OpenAPI specification.
- **Learning:** Whether you're trying to get documentation for a third party API based on traffic, or are trying to switch to design-first at an organization with no OpenAPI at all, learning can help you move your API spec forward and keep it up to date.
- **Miscellaneous:** Anything else that does stuff with OpenAPI but hasn't quite got enough to warrant its own category.
- **Mock Servers:** Fake servers that take description document as input, then route incoming HTTP requests to example responses or dynamically generates examples.
- **Parsers:** Loads and read OpenAPI descriptions, so you can work with them programmatically.
- **SDK Generators:** Generate code to give to consumers, to help them avoid interacting at a HTTP level.
- **Security:** By poking around your OpenAPI description, some tools can look out for attack vectors you might not have noticed.
- **Server Implementations:** Easily create and implement resources and routes for your APIs.
- **Testing:** Quickly execute API requests and validate responses on the fly through command line or GUI interfaces.
- **Text Editors:** Text editors give you visual feedback whilst you write OpenAPI, so you can see what docs might look like.

Converters

Various tools to convert to and from OpenAPI and other API description formats.

Name	Language	v3.1	v3.0	v2.0	GitHub
Apimatic Transformer - Transform API Descriptions to and from RAML, API Blueprint, OAI v2/v3, WSDL, etc.	SaaS	✗	✓	✓	



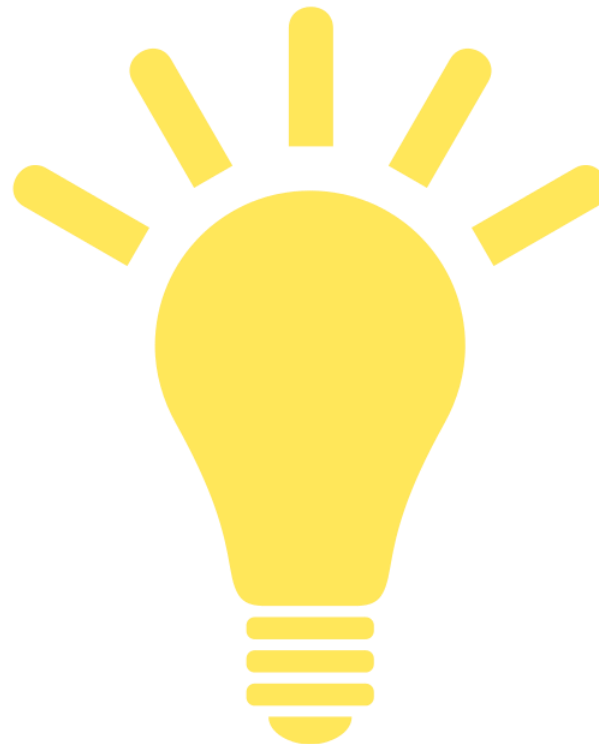
THE
DEVELOPER'S
CONFERENCE

Unindo os 2 mundos

Quero a automação e qualidade do
Contrac First.

Mas quero que os devs não
alteram seu modelo de trabalho,
que eles não sabotem a
metodologia.

Com o pipeline mais as
ferramentas do OpenApi posso
alcançar esse objetivo.



Template de API

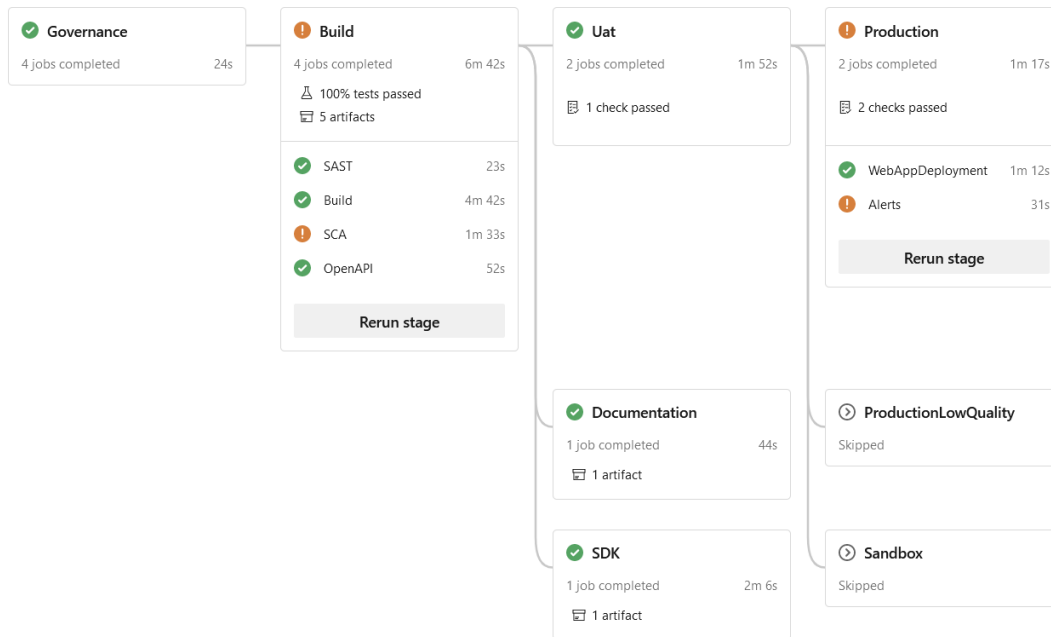
- Tem as dependências que eu quero.
- Posso dar o exemplo para o desenvolvedor.
- Consigo criar *lints* para o dev seguir antes de fazer um *commit*.
- Ajuda na aceleração do desenvolvimento.

```
/// <summary>
/// Lista de clientes.
/// </summary>
/// <response code="200">Retorna lista de endereços.</response>
/// <response code="400">Erro de requisição.</response>
/// <response code="401">Acesso negado.</response>
/// <response code="500">Erro interno da API.</response>
[ProducesResponseType(typeof(IEnumerable<CustomerAddressViewModel>), 200)]
[ProducesResponseType(typeof(ProblemDetails), 400)]
[ProducesResponseType(typeof(ProblemDetails), 401)]
[ProducesResponseType(typeof(ProblemDetails), 500)]
[HttpGet]
public async Task<ActionResult<IEnumerable<CustomerAddressViewModel>>> GetAll()
{
    return Ok(await _customerService.GetAllAsync());
}
```



Etapas do pipeline

- Governança
 - Garantir as informações mínimas.
- Build
 - SAST
 - DevSkim
 - Sonar
 - Build (+Tests)
 - SCA
 - Dependency Track
 - OpenAPI
 - NSwag
- Documentação
 - openapi-generator
- SDK
 - Nswag
- UAT | Sandbox | Production



OpenAPI: automação



THE
DEVELOPER'S
CONFERENCE

```
task: PowerShell@2
name: NSwag
displayName: Run NSwag
inputs:
  targetType: 'inline'
  script: |
    dir ../bin

    $dll = Get-ChildItem -Path ../bin | Where-Object -FilterScript {$_ .Name -match "Wiz.*.API.dll"} | Select-Object -Property FullName, Name -First 1
    Write-Output "$($dll.FullName)"

    $fileExists = Test-Path -Path "$($dll.FullName)"

    $proj = $($dll.Name -replace "Wiz.", "").replace ".Api.dll", ""

    Invoke-Expression -Command "sudo nswag run ./integration/jobs/nswag.json /variables:Configuration=Development,DLL=$($dll.FullName),Namespace=Wiz.Api.$($proj).Client,Name=$($proj)Client /runtime:NetCore31"
```


Amarração



THE
DEVELOPER'S
CONFERENCE

```
- task: PowerShell@2
- name: spectral
- displayName: Run Spectral
- env:
- SYSTEM_ACCESSTOKEN:
- Build_Repository_Name: $(Build.Repository.Name)
- System_PullRequest_PullRequestId: $(System.PullRequest.PullRequestId)
- inputs:
- ignoreLASTEXITCODE: true
- targetType: 'inline'
- script: |
- Invoke-Expression -Command "sudo spectral lint $(Build.Repository.LocalPath)/openapi.json -r ../integration/jobs/wiz.spectral.yaml -f junit -o $(Build.Repository.LocalPath)/openapicode.xml"
- Invoke-Expression -Command "sudo spectral lint $(Build.Repository.LocalPath)/openapi.json -r ../integration/jobs/wiz.spectral.yaml -f json -o $(Build.Repository.LocalPath)/openapicode.json"

- $results = Get-Content $(Build.Repository.LocalPath)/openapicode.json
- $results = $results | ConvertFrom-Json
```

wiz.spectral.yaml

Contents History Compare Blame

```
1 extends:
2   - spectral:os
3   - spectral:asyncapi
4
5 rules:
6   wiz-info-title-default-name:
7     description: Title must be different from "My title".
8     type: style
9     given: $info
10    severity: error
11    then:
12      field: title
13      function: pattern
14      functionOptions:
15        notMatch: "/My title/g"
16
17   wiz-info-title-template-name:
18     description: Title must be different from "Whitelabel API".
19     type: style
20     given: $info
21     severity: error
22     then:
23       field: title
24       function: pattern
25       functionOptions:
26         notMatch: "/Whitelabel API/g"
27
28   wiz-info-description-template:
29     description: Title must be different from "API de Whitelabel".
30     type: style
31     given: $info
32     severity: error
33     then:
34       field: description
35       function: pattern
36       functionOptions:
37         notMatch: "/API de Whitelabel/g"
38
```

Doc: automação



THE
DEVELOPER'S
CONFERENCE

```
--script: |  
| sudo openapi-generator-cli generate -i $(Pipeline.Workspace)/openapi.json -g markdown -o $(Build.Repository.LocalPath)/html  
| displayName: Generate documentation  
| workingDirectory: $(System.DefaultWorkingDirectory)
```



OptInApi

All URLs are relative to <http://localhost>

Method	HTTP request	Description
optInAdd	POST /api/v1/pessoa	Adiciona um novo cliente
optInForget	DELETE /api/v1/pessoa/cpf/{cpf}	Opção pelo direito de esquecimento do cliente
optInGetOptions	GET /api/v1/pessoa/cpf/{cpf}	Busca informações de opt-in de um cliente pelo cpf
optInUpdate	PUT /api/v1/pessoa/cpf/{cpf}/optin	Atualiza informações de opt-in do cliente
optInUpdateContact	PUT /api/v1/pessoa/cpf/{cpf}/contact	Atualiza a forma de contato preferencial do usuário

optInAdd

PersonOptions optInAdd(personOptions)

Adiciona um novo cliente

Parameters

Name	Type	Description	Notes
personOptions	PersonOptionsViewModel	Parâmetro com as informações do cliente a serem gravadas	

SDK: automação



THE
DEVELOPER'S
CONFERENCE

```
--task: PowerShell@2
--name: NSwag
--displayName: Run NSwag
--inputs:
--targetType: 'inline'
--script: |
--dir: ../bin

--$dll = Get-ChildItem -Path ../bin | Where-Object -FilterScript {$_.Name -match "Wiz.*.API.dll"} | Select-Object -Property FullName, Name -f 1
--Write-Output "$($dll.FullName)"

--$fileExists = Test-Path -Path "$($dll.FullName)"

--$proj = $($dll.Name -replace "Wiz.", "" -replace ".Api.dll", "")

--Invoke-Expression -Command "sudo nswag run ./integration/jobs/nswag.json /variables:Configuration=Development,DLL=$($dll.FullName),Namespace=Wiz.Api.$($proj).Client,Name=$($proj)Client /runtime:NetCore31"
```

```
HttpClient httpClient = new HttpClient();

ClientCredentials config = new ClientCredentials(
    baseAuthUrl: "https://sso-hml-web.azurewebsites.net",
    clientId: "ApiOptIn",
    clientSecret: "SHQzHdoUXk-----3NuIEw22",
    scope: "api-optin.fullaccess api-optin");

OptInClient client = new OptInClient("https://optin-hml-api.azurewebsites.net", httpClient, config);

PersonOptions personOptionGet = await client.GetOptionsAsync("02340061105");
Console.WriteLine(personOptionGet.ToJson());
```

wizolucos / WizCross / Artifacts / Packages

You can now search for packages across feeds. Try searching for a package in the search box. [Learn more.](#)

WizCross + Create feed Connect to feed Recycle Bin

Package	Views	Last pushed	Description
 Wiz.Api.OptIn.Client Version 1.0.0.0.0.0		27m ago	SDK para consumir a API OptIn.
 Wiz.Common.Uri Version 1.0.0.0.0.0		19 de set. de 2...	Uma API .NET Standard para f...
 Wiz.Multitenant.Core Version 1.0.0.0.0.0		27 de mai. de 2...	Package Description

Frontend

```
# Usando NPM
npm i -g @stoplight/prism-cli

# Usando Yarn
yarn add global @stoplight/prism-cli
```

```
prism mock -d http://localhost:5001/swagger/v1/swagger.json
```

```
bash-3.2$
```



THE
DEVELOPER'S
CONFERENCE

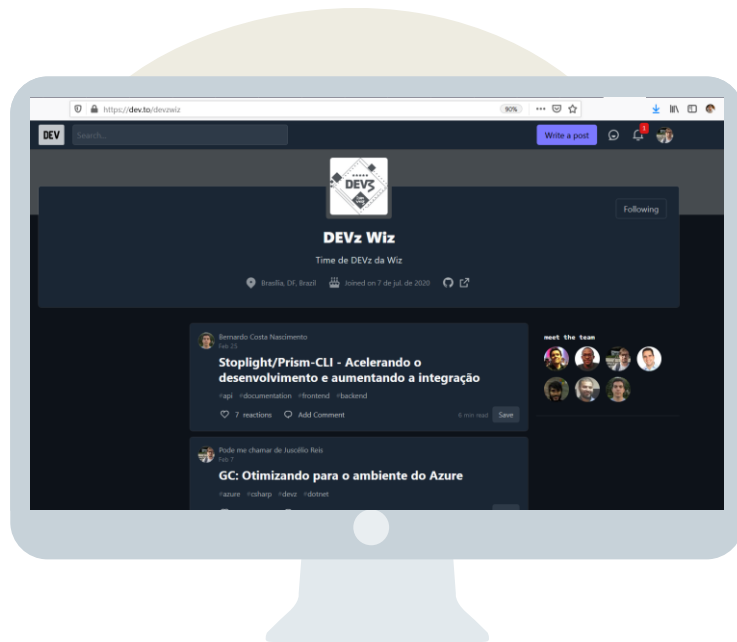
```
namespace Wiz.PrismAPI.API.ViewModels.Customer
{
    /// <example>
    /// { "id": 1, "addressId": 1, "name": "John Doe", "dateCreated": "2021-02-23T17:46:37.364Z", "cep": "35362971", "address": { "id":
    /// </example>
    16 references | Bernardo Costa Nascimento, 4 day ago | 1 author (Bernardo Costa Nascimento)
    public class CustomerAddressViewModel
    {
        [JsonConstructor]
        2 references
        public CustomerAddressViewModel(int id, int addressId, string name, DateTime dateCreated, string cep, AddressViewModel address)
        {
            Id = id;
            AddressId = addressId;
            Name = name;
            DateCreated = dateCreated;
            CEP = cep;
            Address = address ?? new AddressViewModel(addressId, cep,null,null,null);
        }

        [JsonSchemaExtensionData("minimum", 1)]
        [JsonSchemaExtensionData("x-faker", "random.number")]
        1 reference
        public int Id { get; set; }
        [JsonSchemaExtensionData("minimum", 1)]
        [JsonSchemaExtensionData("x-faker", "random.number")]
        1 reference
        public int AddressId { get; set; }
        [JsonSchemaExtensionData("x-faker", "name.findName")]
        1 reference
        public string Name { get; set; }
        1 reference
        public DateTime DateCreated { get; set; }
        [JsonSchemaExtensionData("x-faker", "address.zipCode")]
        1 reference
        public string CEP { get; set; }
        1 reference
        public AddressViewModel Address { get; set; }
    }
}
```

Mais informação



<https://dev.to/devzwiz>
<https://github.com/wizsolucoes>





THE DEVELOPER'S CONFERENCE