

# Importar e exportar projetos

As aplicações desenvolvidas no Cronapp não possuem lock-in e podem ser exportadas para serem desenvolvidas em outras plataformas, pois o conteúdo é extraído em código padrão, sem componentes proprietários, de forma organizada, otimizada e atualizada em linguagem padrão de mercado.

Veremos abaixo como **importar** e **exportar** os projetos e recursos desenvolvidos no Cronapp.

## Importar

Para importar um projeto ou recurso, acesse no menu do sistema **Projeto > Importar** (Figura 1). Essa janela também pode ser acessada através do menu de contexto na raiz do projeto (nome do projeto) na árvore de arquivos.

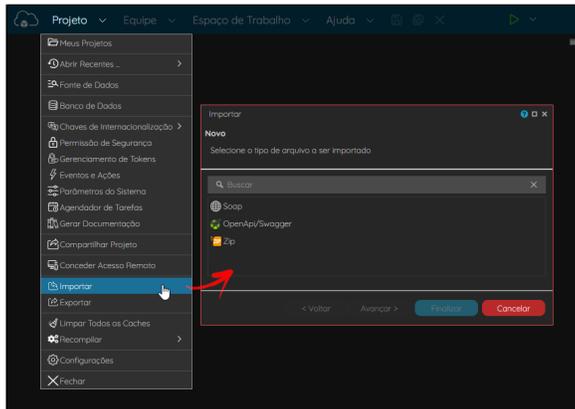


Figura 1 - Opções para importar projetos

- **Soap:** o web service SOAP (Simple Object Access Protocol) é utilizado para disponibilizar serviços interativos e possibilitar a comunicação entre diferentes aplicações através do protocolo SOAP. Através dessa opção será possível gerar **Blocos de programação** desses recursos.
- **OpenApi/Swagger:** uma OpenAPI, ou Interface de Programação de Aplicações Aberta, é uma especificação e conjunto de regras que permitem que diferentes softwares se comuniquem entre si de maneira padronizada. Através dessa opção é possível utilizar uma especificação OpenAPI para gerar **Blocos de programação** e **Fonte de dados** desses recursos.
- **Zip:** permite importar um projeto compactado em um arquivo \*.zip.

## Soap

Abra a janela **Importar**, conforme a Figura 1. Na nova janela (Figura 1.1), selecione a opção **Soap** e clique em **Finalizar**. Preencha as informações na janela **Importar SOAP** e clique em **Importar** ao final.

Acesse a documentação [Consumindo Web Service SOAP](#) para detalhes sobre os campos da janela **Importar SOAP**.

## Nesta página

- **Importar**
  - Soap
  - Open API / Swagger
  - Zip
- **Exportar**
  - Exportar para repositório Git
  - Biblioteca
  - Zip
  - Pacote para Deploy
    - Dockerfile
      - Certifica do SSL
    - Deploy independente
    - Banco H2
    - Exportando war via comando (High-code)

## Saiba mais

- [Criar projeto](#)
- [Abrir projeto](#)
- [Configurações do projeto](#)
- [Estrutura de arquivos](#)

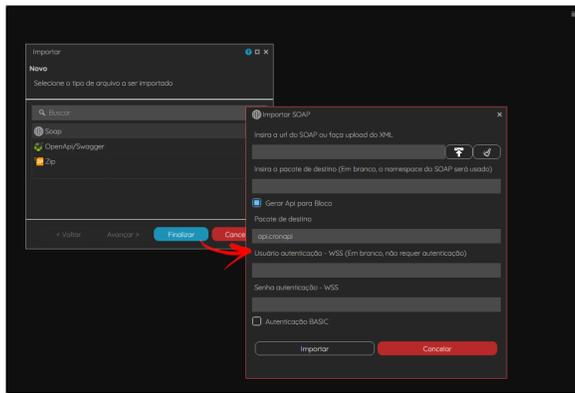


Figura 1.1 - Importar web service SOAP

## Open API / Swagger

Abra a janela **Importar**, conforme a Figura 1. Na nova janela (Figura 1.2), selecione a opção **OpenAPI /Swagger** e clique em **Avançar**. Preencha as informações na janela **OpenAPI** e clique em **Avançar**.

Acesse a documentação [Importar OpenAPI/Swagger](#) para mais detalhes sobre os campos da janela **OpenAPI**.

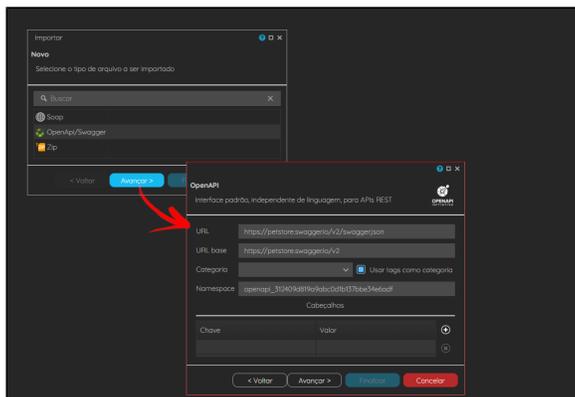


Figura 1.2 - Importar OpenAPI/Swagger

## Zip

Abra a janela **Importar**, conforme a Figura 1. Na nova janela (Figura 1.3), selecione a opção **Zip** e clique em **Finalizar**. Clique em **Selecionar** para escolher o arquivo em seu sistema operacional, espere carregar e clique em **OK**.

Ao importar um projeto Zip a partir de um projeto aberto, o conteúdo do projeto atual será substituído pelo conteúdo do arquivo.

Recomendamos importar um projeto Zip selecionando a opção "O projeto está em um arquivo ZIP?" ao [Criar projeto](#).

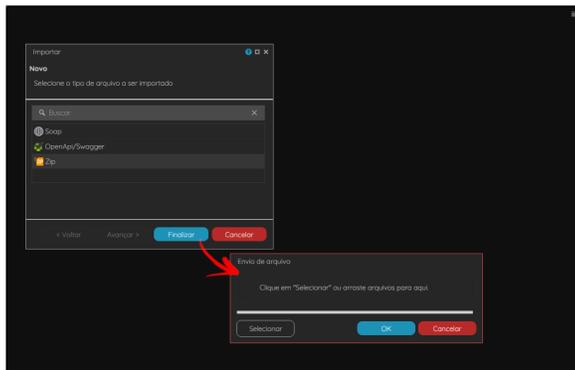


Figura 1.3 - Importar arquivo .zip

## Exportar

Clique com o botão direito do *mouse* sobre o **nome do projeto** na árvore de arquivos e selecione a opção **Exportar** (Figura 2).

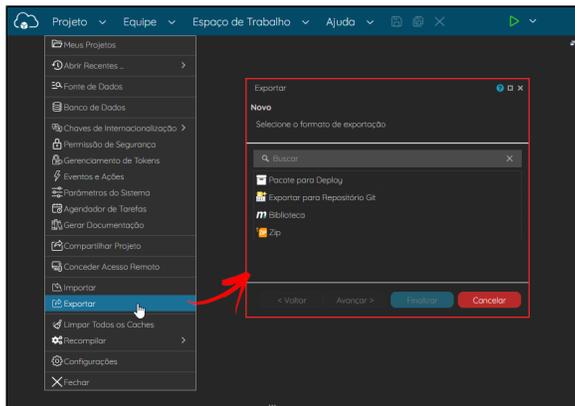


Figura 2 - Opções para exportar projetos

- **Pacote para deploy:** gera pacotes para publicação em um servidor de aplicação Java web, como o Apache Tomcat.
- **Exportar para repositório Git:** o repositório Git é um sistema de controle de versões distribuído. Versionar um projeto com o Git é uma boa prática no desenvolvimento de software, principalmente no desenvolvimento colaborativo.
- **Biblioteca:** a funcionalidade de biblioteca permite fazer uma cópia de alguns recursos selecionados de um projeto Cronapp e utilizá-lo, de forma totalmente independente, em outro projeto, dentro ou fora da plataforma Cronapp.
- **Zip:** essa opção compacta todo o conteúdo do projeto em um único arquivo \*.zip, mantendo a mesma estrutura de arquivos.

## Exportar para repositório Git

Abra a janela **Exportar**, conforme a Figura 2. Na nova janela (Figura 2.1), selecione a opção **Exportar para repositório Git**, clique em **Avançar**, preencha as informações da janela **Versionar o projeto para um repositório Git** e clique em **Finalizar**. Aguarde o progresso e verifique no seu repositório se o projeto foi exportado.

Acesse o tópico "Exportar projetos" da documentação [Versionamento usando Git](#) para detalhes sobre os campos da janela **Versionar o projeto para um repositório Git**.

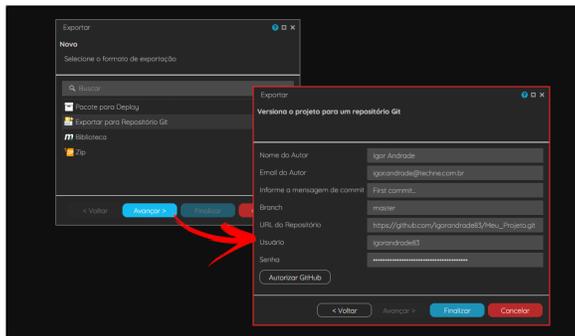


Figura 2.1 - Exportar para repositório Git

## Biblioteca

Abra a janela **Exportar**, conforme a Figura 2. Na nova janela (Figura 2.2), selecione a opção **Biblioteca** e clique em **Finalizar**.

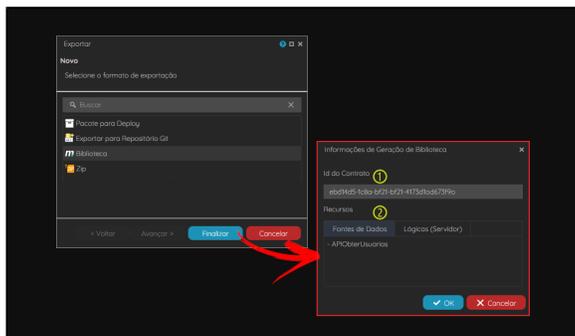


Figura 2.2 - Exportar Biblioteca

Na janela **Informações de Geração de Biblioteca**, confira os dados apresentados e clique em **OK** para iniciar a compilação da biblioteca e, depois de algum tempo, iniciará automaticamente o download do arquivo \*.jar. Acesse a documentação [Importar e exportar bibliotecas](#) para mais detalhes.

1. **Id do Contrato**: identificador do seu contrato dentro do Cronapp.
2. **Recursos**: exibe abas com as **Fontes de dados** e **Blocos de programação** servidor marcados com a opção **Exportar Biblioteca**.

## Zip

Abra a janela **Exportar**, conforme a Figura 2. Na nova janela, selecione a opção **Zip** e clique em **Finalizar**. Após o processo, o download será feito automaticamente. Essa função também pode ser executada através da opção **Baixar** do menu de contexto na raiz do projeto (nome do projeto) na árvore de arquivos.

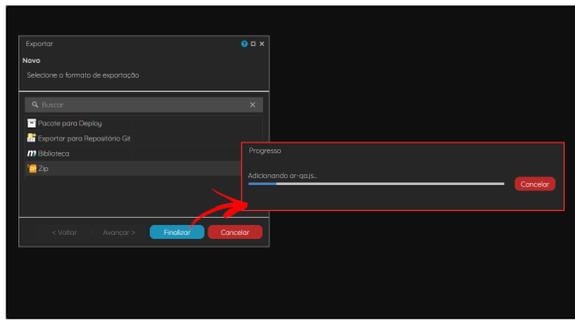


Figura 2.3 - Exportar para repositório Zip

## Pacote para Deploy

Abra a opção **Exportar**, conforme a Figura 2. Na nova janela (Figura 2.4), selecione a opção **Pacote para Deploy** e clique em **Finalizar** para abrir a janela **Opções da Geração de Pacote para Deploy**, configure suas propriedades, clique em **OK** e aguarde o processamento. O download será feito automaticamente.

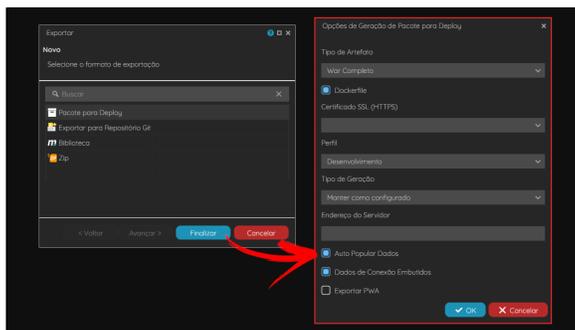


Figura 2.4 - Exportar Pacote de deploy do projeto

### Janela Opções de Geração de Pacote para Deploy:

- **Tipo de Artefato:** define o formato do artefato gerado.
  - O front-end mobile só será incluído no pacote ao selecionar a opção **Exportar PWA**.
  - **War Completo:** arquivo \*.war contendo todo o projeto (back-end e front-end).
  - **War Servidor:** arquivo \*.war contendo apenas o back-end do projeto.
  - **War Front-end:** arquivo \*.war contendo apenas o front-end do projeto.
  - **Zip Front-end:** arquivo \*.zip contendo os arquivos estáticos do front-end.
- Essa opção pode ser utilizada para publicar sem a necessidade de um servidor de aplicação como o tomcat, servindo diretamente ao nginx, por exemplo.
- **Dockerfile:** inclui no pacote o arquivo Dockerfile.
- **Certificado SSL (HTTPS):** permite selecionar um certificado Let's Encrypts gerado e criado no **Serviços de Cloud**. Esse campo ficará ativo apenas ao habilitar a opção "Dockerfile".
- **Perfil:** perfis dos bancos de dados usados no desenvolvimento do projeto: Desenvolvimento, Produção ou outro criado na ferramenta de **banco de dados**.
- **Tipo de Geração:** define as ações que serão tomadas com o banco de dados.
  - **Manter como configurado:** mantém as mesmas configurações aplicadas no momento da geração da persistência no Diagrama de Classe;
  - **Criar ou Atualizar Tabelas:** use essa opção para gerar as tabelas do banco de dados pela primeira vez ou caso tenha adicionado novas classes e não tenha feito alterações nas que já existem. Essa opção não apaga os dados do banco de dados;
  - **Apagar e Recriar Tabelas:** apaga todas as tabelas, incluindo seus dados, do banco de dados e recria tudo novamente com as novas alterações feitas;
  - **Utilizar Tabelas Existentes:** não realiza alterações no banco de dados;
- **Endereço do Servidor:** esse campo é obrigatório ao exportar artefatos contendo apenas o front-end, ele espera receber a URL (domain) da aplicação servidor do projeto.

- **Auto Popular Dados:** após criar as tabelas do banco de dados, os dados contidos no arquivo `populate.json` serão inseridos automaticamente.
- **Dados de Conexão Embutidos:** ao desmarcar, tanto os dados de conexão com o [banco de dados](#) quanto os [parâmetros do sistema](#) configurados não serão incluídos no arquivo `*.war`, sendo necessário configurá-los posteriormente no servidor da aplicação.
- **Exportar PWA ou Incluir Projeto Mobile:** inclui no pacote `*.war` o diretório `mobileapp/` com o conteúdo do projeto mobile. Esse recurso pode ser utilizado para o PWA. Mais detalhes na documentação [Cronapp PWA](#).

## Dockerfile

Ao selecionar essa opção na janela de **Opções de Geração de Pacote para Deploy**, o pacote sofrerá algumas alterações:

- O pacote virá compactado em um arquivo `*.zip`.
- O arquivo `Dockerfile` será incluído dentro do arquivo compactado, junto com o artefato, e seu conteúdo será modificado com base nas opções selecionadas.
- Selecionando o tipo de artefato **Front-end zip**, o Tomcat não será necessário, podendo servir apenas ao Nginx.
- Os arquivos `run.bat` e `run.sh` serão incluídos dentro do arquivo compactado, junto com o artefato. Eles possuem comandos docker para simplificar ainda mais a virtualização da aplicação.

### Executando a aplicação no Linux

```
bash run.sh
```

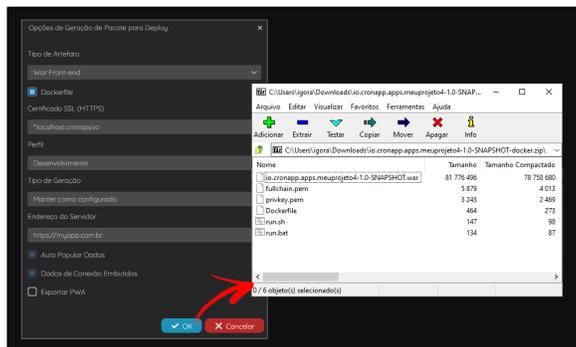


Figura 2.5 - Configuração do pacote e o conteúdo compactado aberto após o download

## Certificado SSL

Essa opção ficará ativa após habilitar a opção **Dockerfile** e permitirá a seleção de um dos certificados Let's Encrypts gerado e criado no [Serviços de Cloud](#). Com isso, o pacote sofrerá algumas alterações:

- Os arquivos `fullchain.pem` e `privkey.pem` serão incluídos dentro do arquivo compactado, junto com o artefato.
- O `Dockerfile` será modificado e apontará para os caminhos dos arquivos `*.pem`.
- O Nginx será configurado para atender HTTPS, servindo na porta 443 e redirecionando a porta 80 para ela.
  - Se não for selecionado um certificado, o Nginx atenderá HTTP e servirá apenas na porta 80.

## Deploy independente

Ao separar os artefatos do projeto em back-end e front-end, será possível manter duas publicações distintas se comunicando. Para isso, é necessário atender aos critérios abaixo:

- O back-end e front-end devem ser exportados de um mesmo projeto Cronapp.
- Durante a criação do front-end, é necessário informar a URL (domain) do back-end da aplicação.

Projetos Cronapp criados antes da versão 2.9.6-SP.44 precisam que o arquivo `hostApp.js` seja referenciado como script no `index.html` manualmente, tanto na aplicação web quanto mobile. Dessa forma, acesse o menu de contexto do arquivo **hostApp.js** (Endereço web: `src/main/webapp/js/hostApp.js` | Endereço mobile: `src/main/mobileapp/www/js/hostApp.js`) e selecione a opção **Importar dependência**.

## Banco H2

Caso o Perfil selecionado contenha um banco de dados local (H2), mesmo que não utilizado, e a opção **Dados de Conexão Embutidos** esteja marcada, uma mensagem de alerta será exibida. (Figura 2.6)

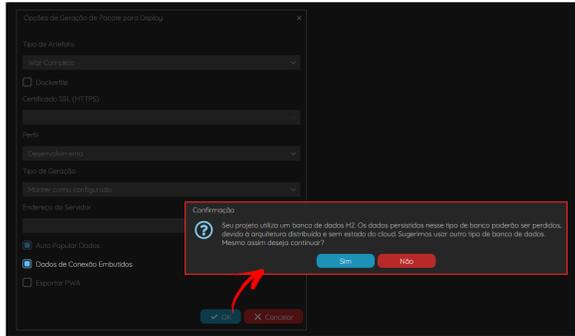


Figura 2.6 - Mensagem de confirmação de uso de banco de dados local

Ao clicar em **Sim**, a IDE irá criar um novo arquivo para o banco H2 no pacote `*.war` da aplicação, não carregando os registros do banco de dados em ambiente de desenvolvimento.

## Exportando war via comando (High-code)

É possível parametrizar as propriedades da janela **Opções de Geração de war** no comando Maven `"mvn package"`. Esses parâmetros são úteis para usuários que automatizam **entregas/integração contínuas** (CD/CI pipeline) através de ferramentas externas, como por exemplo, o Jenkins.

Os parâmetros extras do `"mvn package"` são:

- **-Dcronapp.artifactType**: define o formato do artefato gerado. Deve ser utilizado com uma das chaves abaixo.
  - **all**: arquivo `*.war` contendo todo o projeto (back-end e front-end) (Padrão).
  - **server**: arquivo `*.war` contendo apenas o back-end do projeto.
  - **front-end-war**: arquivo `*.war` contendo apenas o front-end do projeto.
  - **front-end-zip**: arquivo `*.zip` contendo os arquivos estáticos do front-end.
- **-Dcronapp.dockerfile**: adiciona o `Dockerfile` no arquivo compactado, junto com o artefato (true/false - Padrão: true).
- **-Dcronapp.profile**: define o **perfil da aplicação** (DEV, PROD etc - Padrão: DEV).
- **-Dcronapp.generationType**: indica qual o tipo de geração de entidades (Para mais detalhes, acesse a documentação do [eclipseLink](#)). Deve ser utilizado com uma das chaves abaixo.
  - **create-tables**: criar tabelas.
  - **create-or-extend-tables**: criar ou atualizar tabelas.
  - **drop-and-create-tables**: apagar e recriar tabelas.
  - **none**: manter como configurado (Padrão).
- **-Dcronapp.hostApp**: URL (domain) da aplicação servidor, necessário para os pacotes que contém apenas o front-end.
- **-Dcronapp.populate**: indica se deve incluir o `populate.json` (true/false - Padrão: true).
- **-Dcronapp.useContext**: indica se deve incluir o `context.xml` com os dados de conexão (true/false - Padrão: true).
- **-Dcronapp.mobileapp**: indica se deve incluir o projeto mobile (true/false - Padrão: false).
- **-Dcronapp.fullChain**: caminho do certificado (`fullchain.pem`).
- **-Dcronapp.privkey**: caminho da chave privada (`privkey.pem`).
- **-Dcronapp.gzip**: utiliza compactação nas requisições (on/off - Padrão: on).
- **-Dcronapp.timezone**: define o fuso horário do container no servidor da aplicação, veja mais detalhes na lista de **timezones Java** (padrão: America/Fortaleza).
- **-Dcronapp.mem**: percentual de memória usada do container pelo tomcat (0~100 - Padrão: 80).

Cada parâmetro deve ser precedido com `"-D"`, como no exemplo abaixo:

## Parâmetros Cronapp para o Maven

```
mvn package -Dcronapp.profile=DEV -Dcronapp.populate=true -Dcronapp.useContext=true -Dcronapp.generationType=create-tables -Dcronapp.mobileapp=true -Dcronapp.artifactType=all -Dcronapp.dockerfile=true
```

Para acessar a janela do terminal, clique no **botão do menu do sistema** (destaque 1 da figura 2.7) e em seguida clique no ícone do **Terminal** (2).

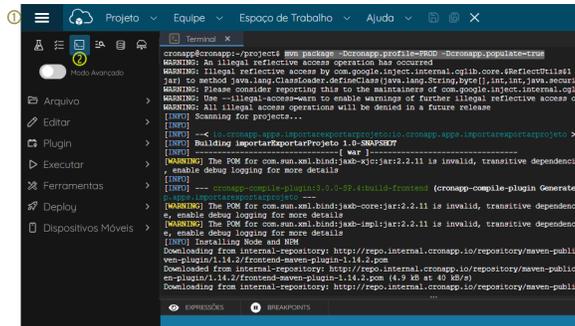


Figura 2.7 - Exportar o arquivo .war do projeto por linha de comando

No terminal, o comando deve ser executado a partir do diretório `~/project/` e o pacote gerado será encaminhado para o diretório `~/project/target/`, ficando acessível apenas via terminal.