TEMPLATE ETLIPSE REVIT ADDIN WPF VISUAL STUDIO – INSTRUÇÕES

Olá, mundo! Estas são algumas instruções que você deve seguir para utilizar o nosso template de Visual Studio para add-in de Revit com múltiplos comandos em *modeless dialog*.
(faça o download aqui)

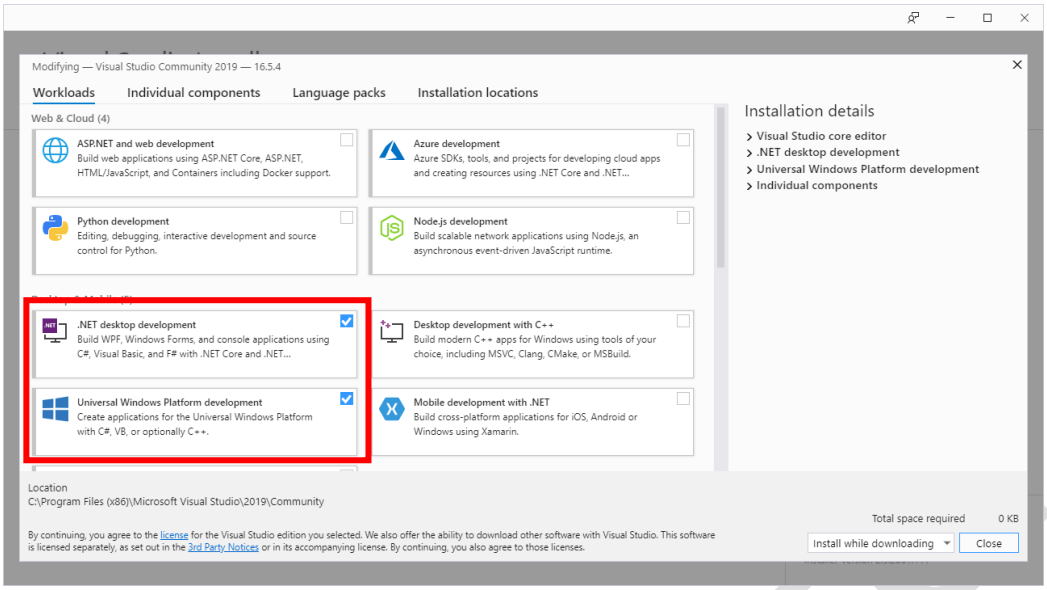
É importante observar que este é um trabalho baseado e desenvolvido a partir de estudos sobre o projeto de exemplo para Windows-Forms contido no *Revit SDK* chamado *ModelessForm_ExternalEvent*. Então desenvolvemos uma solução que adapta a lógica do exemplo a um simples e prático projeto de aplicação em WPF, que pode ser usado como template.

O Código no template é comentado com bastante explanação sobre cada classe, método e interação utilizados no projeto. Abaixo você tem as instruções e algumas dicas para solucionar problemas ao final deste documento. Vamos lá!

Instruções Gerais

1. O primeiro item necessário, é claro, é uma cópia do Autodesk Revit 2019 ou 2020 instalada no seu computador. Apesar de não termos testado nosso template para outras versões, se este for o seu caso, você pode ver os passos na última seção deste documento, *Solução de Problemas*.
2. O Segundo item necessário é o Microsoft Visual Studio instalado em seu computador. Nosso template é feito para este programa. Se você ainda não possui uma cópia instalada, você pode facilmente baixar a atual versão gratuita, *Visual Studio Community 2019*, aqui, para que você possa praticar tudo que você precisa para criar o seu add-in.
3. Considere que a instalação do Visual Studio pode levar bastante tempo. Mas você não deve enfrentar problemas durante a o processo ou utilizando uma conta da Microsoft. A única decisão importante para a qual você precisa atentar é quanto aos pacotes que você deseja instalar. Para poder utilizar o template, você só precisa marcar as cargas de trabalho *.NET desktop development* e *Universal Windows Platform development* (como na imagem abaixo). Por agora, é suficiente e você poderá finalizar a instalação, mas não esqueça que você pode sempre executar o instalador do Visual Studio novamente caso você precise instalar outros pacotes para suas futuras aplicações.

eTLipse - Computação, Engenharia e Responsabilidade Social Ltda.
www.etlipse.com | info@etlipse.com
Desenvolvido por Joel Diniz e Edson Andrade

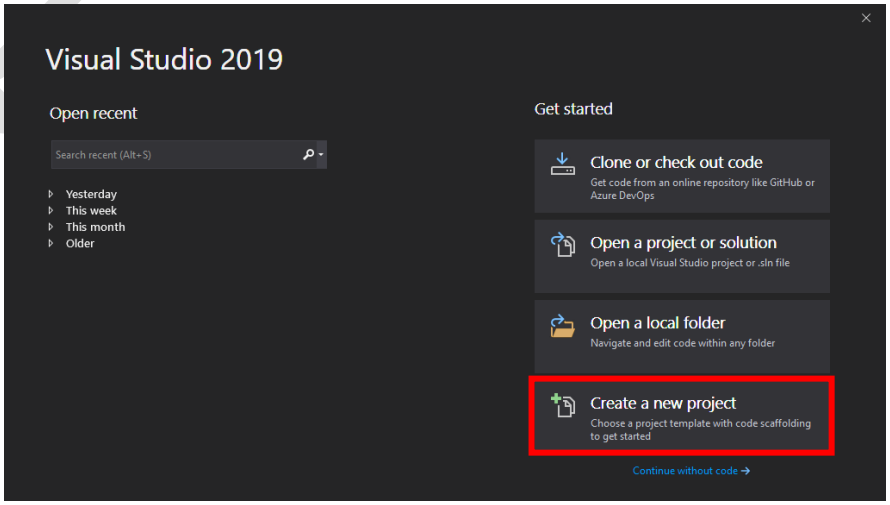


4. Com o Visual Studio instalado, baixe o nosso template aqui. Os arquivos .zip estão nomeados de acordo com a versão do Revit para o qual foram testados. Então, por exemplo, se você possui o Revit 2019, você deve baixar o arquivo *TL Revit 2019 WPF Addin.zip*. Lembre-se que, se você possuir uma versão diferente da 2019 ou 2020, você pode ver o último procedimento na seção *Solução de Problemas* no final do documento.
5. Agora é hora de colocar o template no lugar adequado. Copie o arquivo .zip que você baixou para a seguinte pasta:

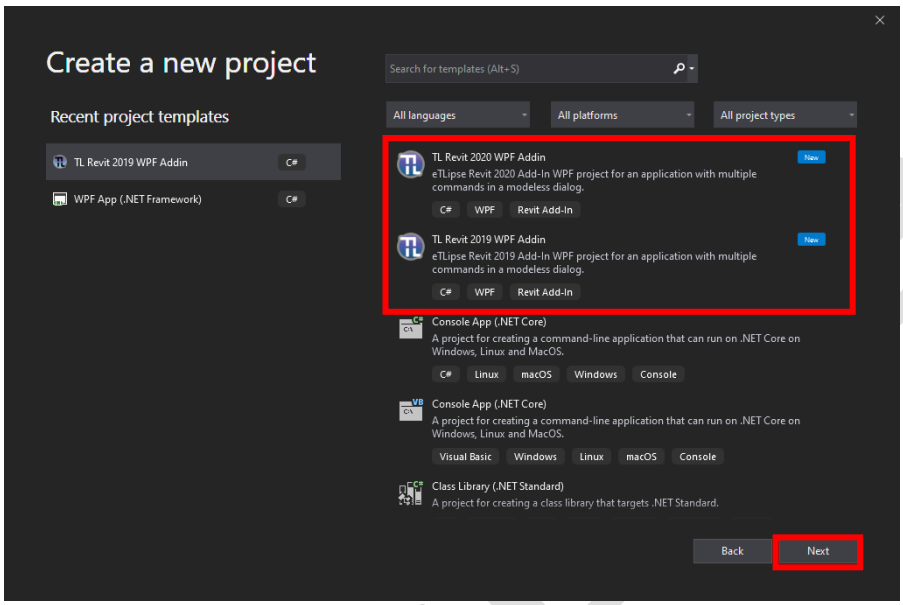
```
C:\Users\<User>\Documents\Visual Studio ****\Templates\ProjectTemplates\
```

Onde "<User>" é sua pasta de usuário no Windows e "****" é a versão do Visual Studio que você possui ("2019", por exemplo).

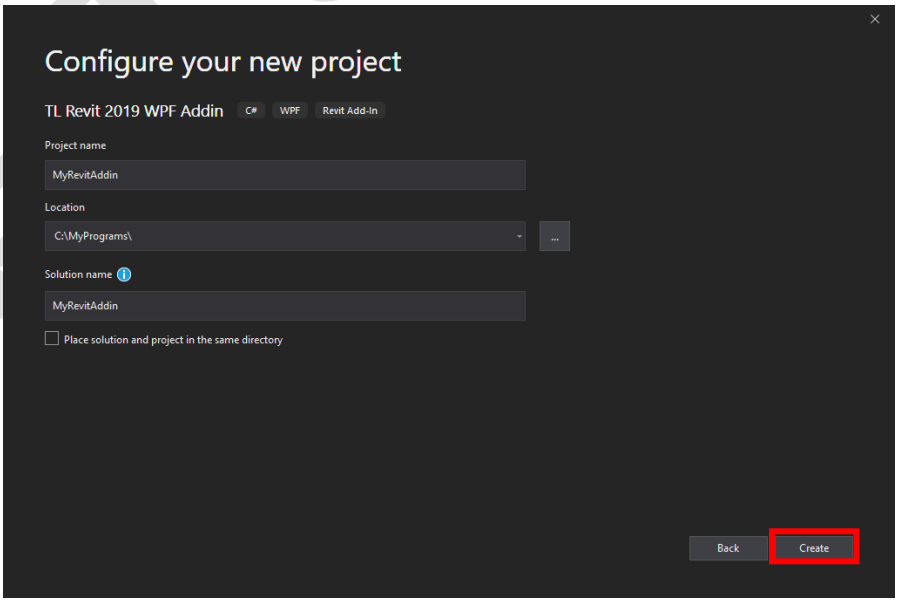
6. Abra o Visual Studio e clique no botão *Create a new project* (imagem abaixo).



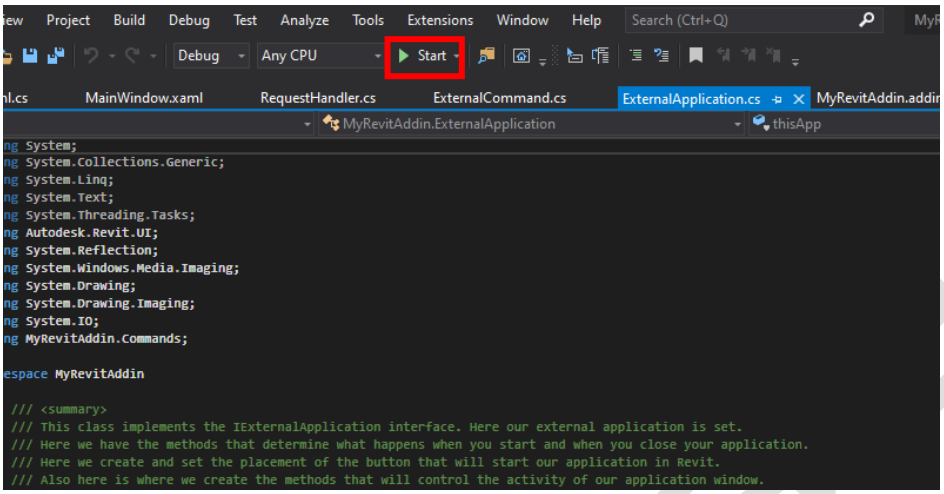
7. Agora, se você copiou o arquivo .zip para o local correto, você deve ver a opção para selecionar o template *TL Revit WPF Addin*, de acordo com sua versão do Revit, e começar o seu projeto (imagem abaixo). Selecione esta opção e clique em *Next*.



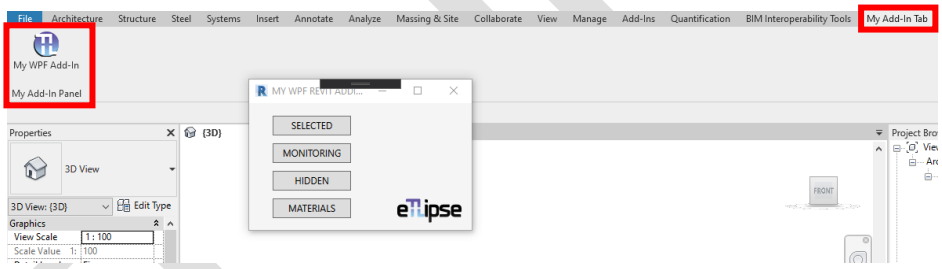
8. Agora você deve nomear o seu projeto (e sua solução), bem como definir o local dos arquivos. Na imagem abaixo você vê apenas um exemplo. Escolha o nome e pasta que você preferir. Quando finalizar, clique em *Create* e seu projeto está configurado.



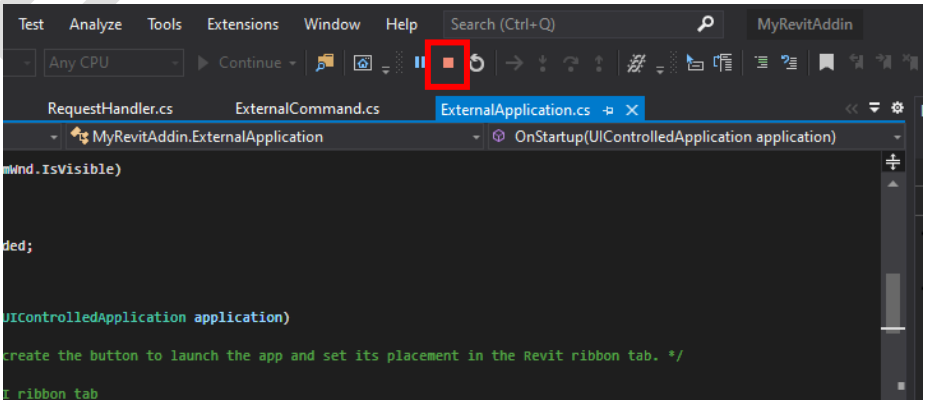
9. Antes de interagir com qualquer item de projeto no template, recomendamos que você simplesmente clique no botão *Start* (imagem abaixo) para compilar e executar o add-in.



10. Uma instância do seu Revit deve se iniciar. Abra qualquer projeto no Revit e então execute o add-in acessando a aba da ribbon *My Add-In Tab*, e então clicando no botão *My WPF Add-In* no painel *My Add-In Panel* (imagem abaixo).



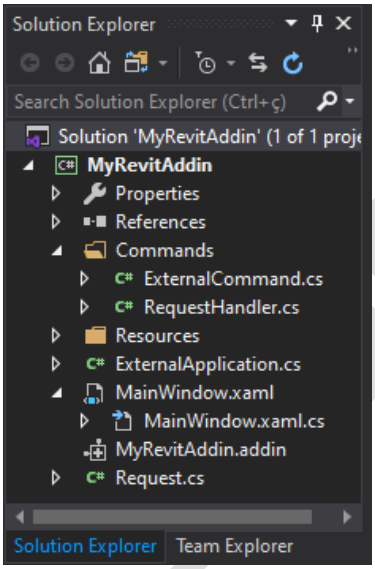
11. Agora, teste os comandos (botões) na interface de usuário do add-in. Quando terminar, volte ao Visual Studio e clique no botão *Stop Debugging* (imagem abaixo).



Se você se deparou com alguma mensagem de erro durante os passos acima, você pode ir diretamente para a seção *Solução de Problemas*, no final deste documento.

Itens, References e Propriedades de Projeto

Seu projeto está aberto no Visual Studio e você tem a seguinte estrutura no seu Solution Explorer:



Properties: aqui você tem configurações importantes do seu projeto. Você define qual plataforma e aplicação serão utilizadas nos seus debugs, para qual local os seus arquivos de saída serão copiados, bem como informação essencial do projeto, como nomes de assembly e namespace.

References: aqui é onde você encontra e adiciona referências como bibliotecas, soluções externas e frameworks que serão necessários para o seu projeto. Duas referências específicas são necessárias para que se possa utilizar os itens, métodos e interações da API do Revit neste template: *RevitAPI.dll* e *RevitAPIUI.dll*. Elas já estão adicionadas ao projeto do template, mas se você encontrar algum erro ou aviso sobre estas duas *dlls*, veja a seção *Solução de Problemas* no final deste documento.

ExternalApplication.cs: código *C#*. Esta classe implementa a interface *IExternalApplication*. Aqui que sua aplicação externa é configurada. Aqui você tem os métodos que determinam o que acontece quando você inicia e quando você fecha sua aplicação. Aqui você cria e configure a localização do botão que irá executar sua aplicação no Revit. Também aqui é onde você cria os métodos que controlarão a atividade da *janela* da sua aplicação.

ExternalCommand.cs: código *C#*. Esta classe implementa a interface *IExternalCommand* e e contém o comando que executa a aplicação, que será chamado pelo botão criado na classe da aplicação externa.

RequestHandler.cs: código *C#*. Esta é a classe que implementa a interface *IExternalEventHandler* para manipular todos os comandos iniciados pela ação do usuário na *MainWindow* (*modeless dialog*) na forma de requisições listadas em uma enumeração utilizada pela classe *Request*. Também aqui você irá definir todos os métodos que irão constituir a funcionalidade da aplicação utilizando a API do Revit.

Request.cs: código C#. Esta é a classe que receberá o comando do usuário por uma enumeração de *RequestId* e fará a requisição quando o *RequestHandler* identifica-lo, enquanto o evento externo referente é levantado.

MainWindow.xaml: código *WPF*. Esta é a janela contendo os botões que irão converter as ações do usuário em requisições para o Revit.

MainWindow.xaml.cs: código *C#*. Este é o código da classe do controle *MainWindow*, que apresenta a principal interface de usuário da aplicação e que queremos que funcione como um *modeless dialog*. Aqui você implementa métodos que devem ser chamados por outras classes para controlar a atividade da janela. Aqui você também define as interações entre a ação do usuário pelos controles da janela e as classes que contêm os comandos que o Revit deve receber como requisições de evento externo.

MyRevitAddin.addin: código *xml*. Este é o arquivo de *manifesto* com referência à classe da aplicação externa e a *dll* de saída do seu add-in. O Revit precisa disto para executar a aplicação. Por padrão em nosso template, tudo será nomeado conforme o nome do projeto.

Com este conhecimento inicial é hora de você brilhar! Nós recomendamos, para iniciantes, ir à classe *RequestHandler* e editar os métodos que você vê na seção *METHODS TO EXECUTE THROUGH DELEGATE* com códigos próprios. Exercite suas habilidades e conhecimento sobre a API do Revit através destes múltiplos comandos. Note que sempre que você clica em *Start* no Visual Studio, antes de executar o Revit, ele irá copiar um arquivo *.pdb*, seu *manifesto (.addin)* e uma *.dll* da sua aplicação na pasta de usuário de *Addins* do Revit na sua máquina. Então você deve apagar estes arquivos antes de todas as vezes em que precisar rodar um novo teste para ver sua solução em ação. A pasta onde você normalmente pode encontrá-los é:

```
C:\Users\<User>\AppData\Roaming\Autodesk\Revit\Addins\****\
```

Onde "<User>" é sua pasta de usuário no Windows e "****" é a versão do Revit que você possui ("2019", por exemplo).

Estaas cópias de arquivos de saída têm seu destino definido em *Properties* no seu projeto no Visual Studio. Se você for até lá, você verá na seção *Build Events* uma linha na caixa de texto *Post-build event command line*. Esta é a linha de commando que envia as cópias dos arquivos no seu Debug para a pasta de usuário de *Addins* do Revit.

Basicamente o fluxo mais seguro é este: você trabalha o seu código> você deleta arquivos antigos na pasta de *Addins*> você executa a solução> você testa no Revit>você para o debug> (repete).

Quando terminar de testar as principais funcionalidades da sua aplicação, você pode explorar o controle *MainWindow* e outras interações no template para desenvolver por completo seu próprio add-in. Esperamos que este template ajude muitas pessoas.

External Links and other References

Há um material extra que também pode ser útil (clique no título de cada item para acessar/baixar):

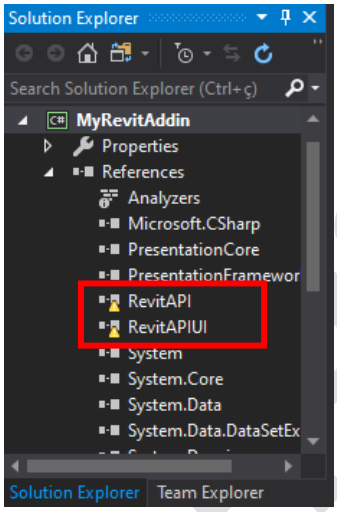
- Se você quer tornar o manejo de seus arquivos de add-in menos repetitivo, dê uma olhada no *Add-In Manager*.
- Nosso guia *Programação Básica de um Add-In para Revit*, uma seção à parte do nosso manual geral (abaixo), concentrando no básico da API do Revit para add-ins.
- Nosso manual *Uso da Programação Computacional na Metodologia BIM* , onde trabalhamos os fundamentos da API do Revit para fluxos de add-ins e *Dynamo*.
- Nosso add-in de Revit *TL SOLIDS Interaction Tool*, útil para operações com sólidos no Revit (União, Corte, Alterar Ordem de União) e checagens (interseções, elementos unidos) com um belo conjunto de ferramentas de visualização e seleção.
- Se você procura por um template útil de add-in para Revit não baseado em WPF, recomendamos o excelente *BIM² Revit Add-In Template* do *James Simpson*.
- E se você procura por publicações e referência de código para estudar a API do Revit para add-ins, recomendamos o blog *The Building Coder* do *Jeremy Tammik* – provavelmente a mais conhecida e renomada referência para conhecimento na API do Revit – e também o repositório no GitHub da *Torsion Tools*, do *Sean Page*, com compartilhamento de muitos códigos providenciais.

Troubleshooting

As dicas vistas nesta seção podem ser de ajuda para resolcer alguns dos problemas que você pode encontrar ao utilizar qualquer template de add-in para Revit no Visual Studio.

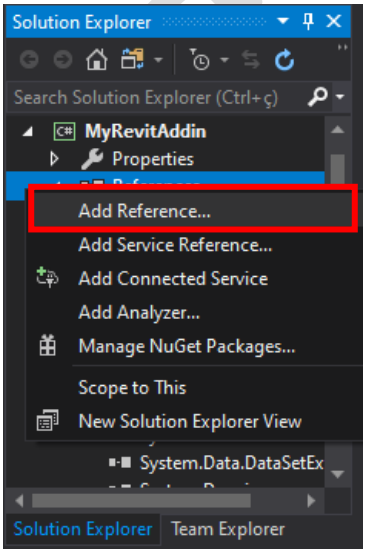
As Referências do Revit precisam ser encontradas

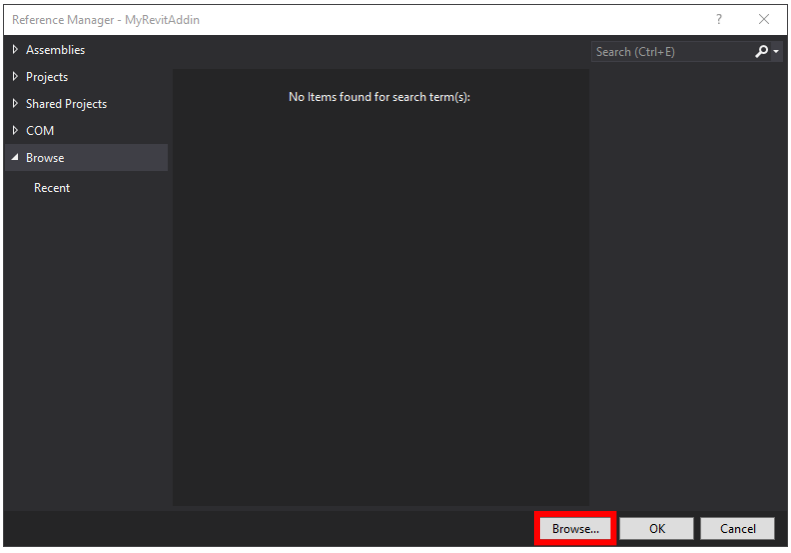
Em alguns casos, você pode se deparar com o seguinte problema nas *References* do seu projeto:



Isto acontece quando seu projeto não consegue encontrar as respectivas *dlls* do Revit. Como consertar isto?

1. Clique com botão direito em *References* e depois em *Add Reference*. Então uma janela é aberta, você pode simplesmente clicar em *Browse* (imagens abaixo).



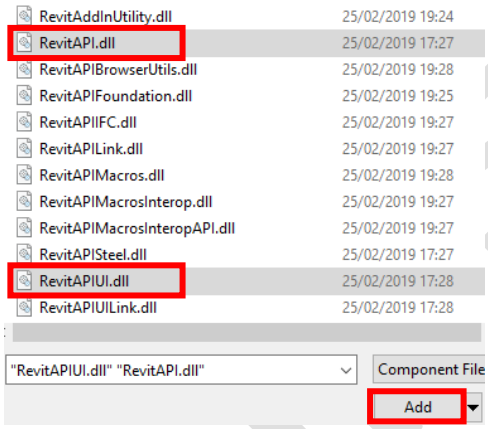


2. Uma nova janela aparece. Acesse a seguinte pasta (ou a pasta onde sua cópia do Revit está instalada):

```
C:\Program Files\Autodesk\Revit ****\
```

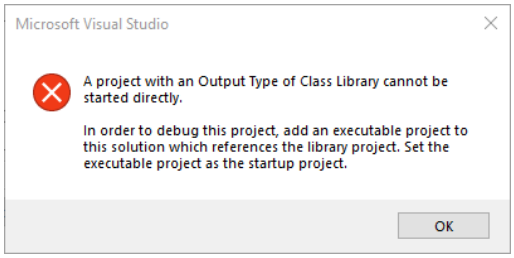
Onde "****" é a versão do Revit que você possui ("2019", por exemplo).

3. Agora selecione tanto o arquivo *RevitAPI.dll* quanto o *RevitAPIUI.dll* (você pode usar a tecla *ctrl* para ajudar com esta tarefa) e clique em *Add* (imagem abaixo), depois em *OK* e você terminou!

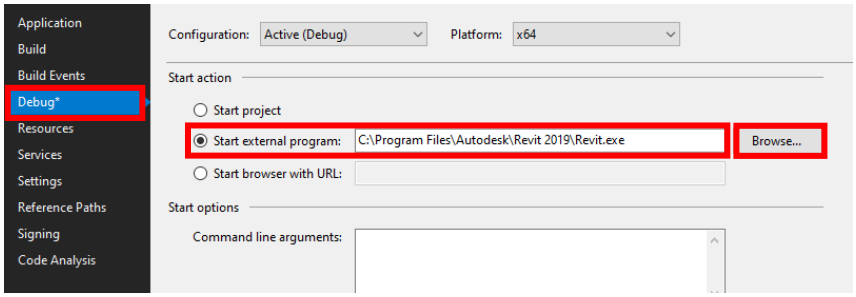


O Revit precisa ser definido como aplicação a iniciar na Depuração (Debug)

Tudo está correto no código, mas você clicou em *Start* no Visual Studio e, em vez de ver a tela de abertura do Revit, você vê apenas isto:

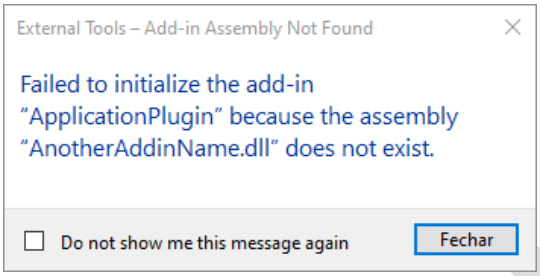


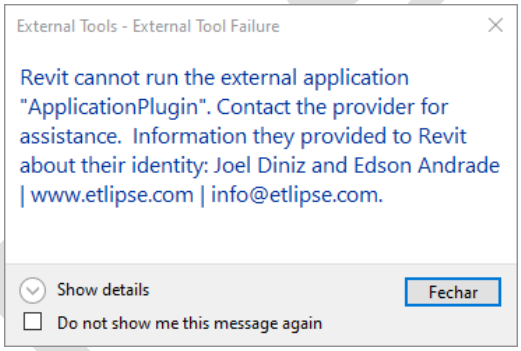
Isto acontece porque o Revit não está indicado como programa externo que iniciará na depuração do código da sua aplicação. Para consertar isto, simplesmente acesse as *Properties* do seu projeto, vá a seção *Debug* à esquerda, marque *Start external program* e busque (Browse) pelo arquivo *Revit.exe* no caminho exibido na imagem abaixo:



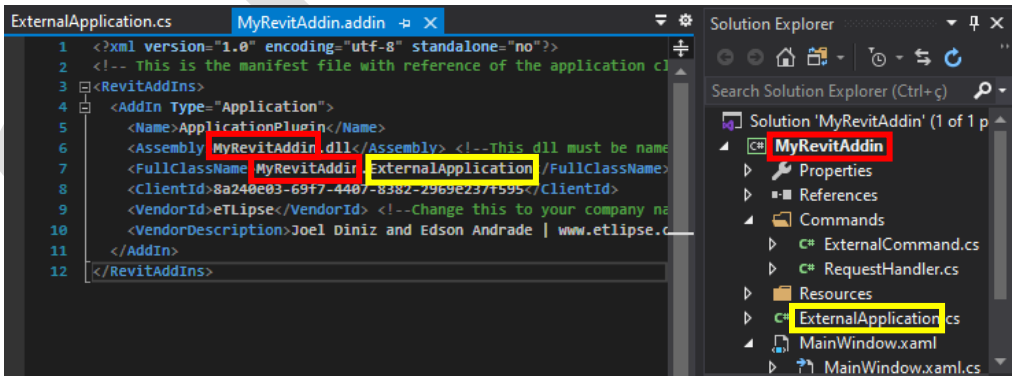
O assembly e a classe da aplicação devem ser referenciados corretamente

Algum dos avisos abaixo aparece quando você tenta testar sua aplicação?





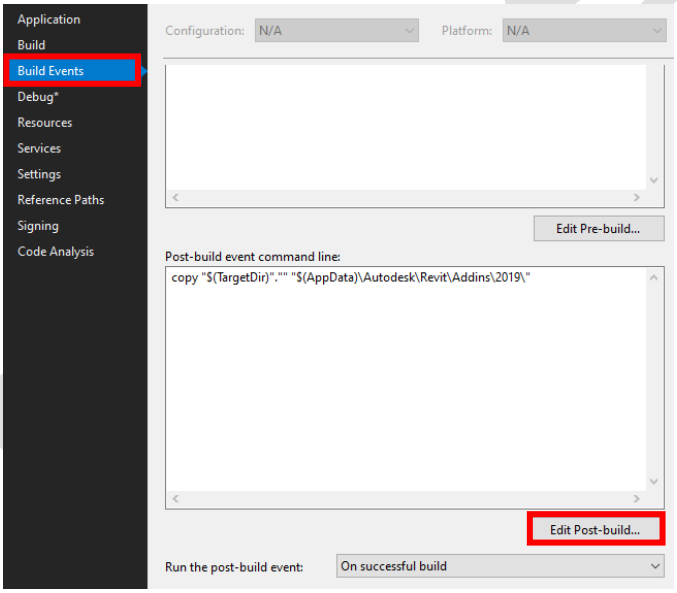
Isto acontece quando o arquivo de manifesto falha em apontar para os arquivos corretos por conta de referência incorreta. Para simplificar as coisas e evitar problemas, recomendamos que você nomeie sua *dll* com o mesmo nome do seu projeto e verifique se o *namespace* da classe da sua aplicação externa está escrito corretamente no arquivo do manifesto (veja imagem abaixo).



Você pode usar este template para outras versões do Revit

Finalmente, se você tem uma versão do Revit diferente da 2019 e da 2020, você pode ainda criar um projeto no Visual Studio a partir deste template e mudar algumas coisas para utilizá-lo:

1. A primeira coisa que você deve fazer é realizar o procedimento de solução de problemas *As Referências do Revit precisam ser encontradas*, descrito acima. Isto irá acessar as *dlls* corretas da pasta do seu Revit.
2. Depois, você deve realizar o procedimento de solução de problemas *O Revit precisa ser definido como aplicação a iniciar na Depuração (Debug),* também descrito acima. Isto fará o Visual Studio inicializar a sua versão do Revit durante a depuração.
3. Os últimos passos consistem em mudar o diretório pós-compilação para os seus arquivos. Ainda em *Properties*, vá à seção *Build Events* e clique no botão *Edit Post-build...* (imagem abaixo).



4. Agora a linha de commando abre e você só precisa substituir o “2019” ou “2020” que você vê na linha pelo número da sua versão do Revit (“2021”, como no exemplo abaixo).