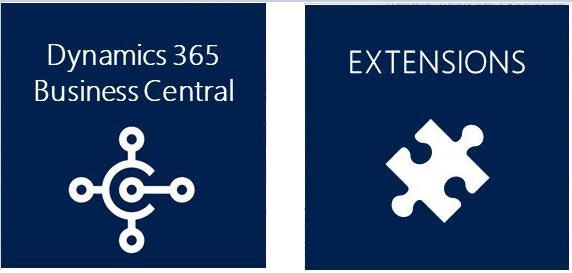
**Exercícios**



**Exercício 1 (**<https://docs.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-extension-example#rewards-extension-overview>**)**

**Instalar: GitLens | GitHistory | AL Variable Helper**

1. Crie uma nova extensão (Customer Reward) que permita a possibilidade do utilizador escolher 3 níveis de prémio (Gold, Silver e Bronze) para o cliente. Para cada nível é atribuído uma percentagem de desconto.

**Dicas:**

* 1. **Criar um novo projeto utilizando o comando *“>AL:Go!”*.**
  2. **Criar uma nova tabela (Reward) com os seguintes campos:**
     1. **Reward ID | Code [30]**
     2. **Description | Text[250] | Usar propriedade *“notblank”***
     3. **Discount Percentage | Decimal | Valores entre 0 a 100 | 2 casas decimais**
  3. **Criar um campo na tabela Customer com o nome Reward ID. Relacionar com a tabela Reward.**
  4. **Criar páginas do tipo list e card para a tabela Reward. (Aplicar propriedade “*ApplicationArea=All”* em todos os campos).**
  5. **Disponibilizar o novo campo Reward ID da tabela “*Customer*” na página “*Customer Card*”. (Aplicar propriedade “*ApplicationArea=All”* em todos os campos).**

1. **Ao preencher o campo Reward ID, validar se o cliente está bloqueado, caso esteja o sistema dará um erro.**
2. **Criar um novo campo na tabela Reward com o nome “Last Modified Date” do tipo Date.**
3. **Desenvolver uma função/método que preencha o campo “Last Modified Date” com a data de sistema ao introduzir, modificar, eliminar ou renomear um registo da tabela.**
4. **Criar o package “>AL:Package” e publicar a extensão “>AL:Publish without debugging”.**
5. **Inicializar o Git com o comando “*>Git:Initialize Repository*” e escolher a pasta do projeto que foi definida inicialmente.**
6. **Aplicar o comando “>git:commit All” e escrever no comentário “Exercice\_1”.**

**Exercício 2**

**Instalar: CRS AL Language Extension**

**Configurar: CRS AL Language Extension | CodeCop | GitIgnore**

1. Criar uma codeunit do tipo “*Install”* (Permite executar a codeunit ao instalar a extensão) com o objetivo de preencher a tabela Reward com os seguintes dados:

|  |  |  |
| --- | --- | --- |
| **Reward ID** | **Description** | **Discount Percentage** |
| Gold | Gold Level | 20% |
| Silver | Silver Level | 10% |
| Bronze | Bronze Level | 5% |

**Dicas:**

* 1. **Aplicar a propriedade *“SubType=Install”.***
  2. **Aplicar o trigger “*OnInstallAppPerCompany()”.***

1. Verificar as diferenças no *“Source Control” (****Ctrl + Shift + G****)*.
2. **Aplicar o comando “>git:commit All” e escrever no comentário “Exercice\_2”.**
3. **Criar o package “>AL:Package” e publicar a extensão “>AL:Publish without debugging”.**

**Exercício 3**

1. Criar um Report do tipo List tendo por base a tabela Reward.

|  |  |  |
| --- | --- | --- |
| **Reward ID** | **Description** | **Discount Percentage** |
| Dados | Dados | Dados |



**Dicas:**

* 1. **Utilizar o snippet *“treport”.***
  2. **Utilizar as propriedades RDLCLayout = ‘nomemapa.rdl’ e DefaultLayout = RDLC**
  3. **Após o código do mapa estar concluído, executar as teclas de atalho *“Ctrl + Shift + B”* para criar o ficheiro RDL. Clicar com o botão direito do rato no .rdl e escolher “Open Externally”.**

**Exercício 4**

**Instalar: Multilanguage Editor (** <https://developer.microsoft.com/pt-pt/windows/develop/multilingual-app-toolkit> **)**

1. Criar o ficheiro das traduções “*XLIFF*”. **Ctrl + Shift + B** pra criar o ficheiro .xlf depois de a. E b.
2. Duplicar o ficheiro gerado pelo sistema, colocar o nome com a tag pt-PT.
3. Aplicar algumas traduções no ficheiro com a ferramenta *“Multilanguage Editor”*.
4. Criar o package “>AL:Package” e publicar a extensão “>AL:Publish without debugging”.

**Dicas:**

1. **Editar o ficheiro *“app.json”* e adicionar o parâmetro *“features:[“TranslationFile”]”***
2. **Criar o package “>AL:Package” para o sistema criar o ficheiro .xlf**
3. **Editar o ficheiro duplicado e alterar o nó *“target-language”* para pt-PT**

**Nota:**

**Sempre que é criado o package o sistema volta a criar o ficheiro de tradução original.**

**Exercício 5**

Pretende-se preencher os campos “*Home Page”, “Address”, “City”, “Post Code”* e *“Country/Region Code”* de forma automática através da execução de um webservice aquando o preenchimento do nome do cliente. O nome do cliente terá de ter um domínio, como por exemplo, **microsoft.com**.

1. **Criar uma codeunit com o snippet *“tcodeunit: (CRS: Method – No UI)”*.**
2. Criar uma função que valide se o nome do cliente tem um do domínio da lista abaixo. Utilizar uma lista de textos para resolver o exercício.
   1. Exemplo de Lista dos Domínios:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| .com | .org | .net | .be | .dk | .de | .nl | .pt | .co.uk | .pl |

1. Criar uma função que obtenha os dados do Web Service.
   1. **URI:** <https://api.fullcontact.com/v3/company.enrich>
   2. **APIKey:** 09jK4s8pl2lJrpnbluWsiM3PZXeDNd0f
2. Criar uma função que preencha os dados devolvidos pelo Web Service.
3. Chamar a função no “*OnAfterValidate*” do campo “*Name*” utilizando um objeto do tipo PageExtension associado à página “*Customer Card*”.
4. Criar o package “>AL:Package” e publicar a extensão “>AL:Publish without debugging”.

**Dicas:**

* 1. **Criar um projeto novo utilizando o comando *“>AL:Go!”***
  2. **Utilizar as seguintes variáveis para a construção da ligação ao Web Service:**
     1. **ContentJson: JsonObject;**
     2. **HttpContent: HttpContent;**
     3. **HttpClient: HttpClient;**
     4. **HttpResponseMessage: HttpResponseMessage;**
  3. **Utilizar as seguintes variáveis para a obtenção dos dados do Web Service:**
     1. **LocationsJson: JsonArray;**
     2. **LocationJson: JsonObject;**
     3. **JsonToken: JsonToken;**
  4. **Docs:** [https://docs.fullcontact.com/#introduction-4](https://docs.fullcontact.com/" \l "introduction-4)

**Exercício 6**

1. Ao introduzir um novo cliente no sistema, o nº telefone deve estar preenchido, caso esteja vazio, o sistema deve emitir uma **notificação** ao utilizador para preencher o campo. Utilizar um evento para resolver o exercício.

**Dicas:**

* 1. **Utilizar a seguinte variável para as notificações:**
     1. **TheNotification: Notification;**

1. Criar o package “>AL:Package” e publicar a extensão “>AL:Publish without debugging”.

**Exercício 7**

1. Utilizar o comando “*>git:*Clone” e utilizar o seguinte endereço <https://github.com/CloudReadySoftware/GameLibrary.git>

<https://github.com/luiscosta86/RewardProject.git>

**Exercício 8**

1. Pretende-se converter um conjunto de objetos C\AL em AL. Fazer o download dos seguintes objetos em C\AL [aqui](https://arquiconsult1.sharepoint.com/:u:/r/sites/intranet/navaddon/Documents/Dicas%20ARQ%20DEV/DicasARQDEV_006_Anexos/Objects%20C_AL.fob?csf=1).
2. Importar os objetos e não efetuar a compilação. (Devem recuperar uma nova base dados Business Central).
3. Fazer o download do ficheiro .ps1 [aqui](https://arquiconsult1.sharepoint.com/:u:/r/sites/intranet/navaddon/Documents/Dicas%20ARQ%20DEV/DicasARQDEV_006_Anexos/Convert_CAL_AL.ps1?csf=1) e editá-lo de forma a conseguir converter os objetos C\AL em AL.
4. Criar um novo projeto com os ficheiros convertidos.
5. Criar o package “>AL:Package” e publicar a extensão “>AL:Publish without debugging”.

**Dicas:**

* 1. **Criar um projeto novo utilizando o comando *“>AL:Go!”***