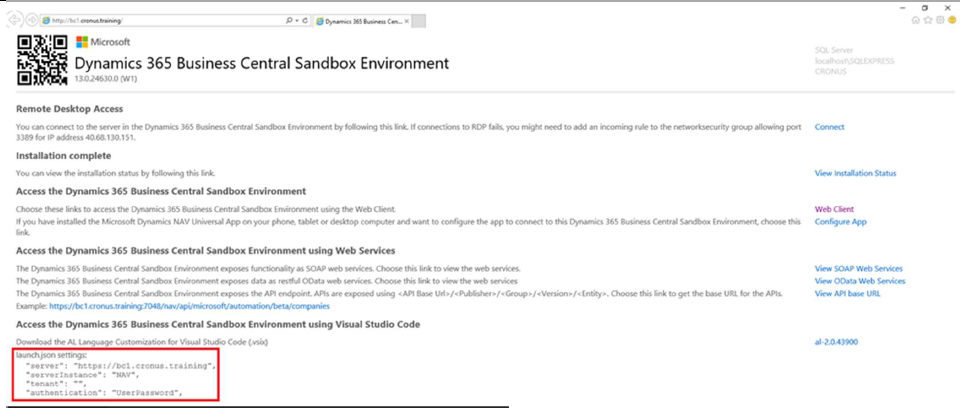
**Data exchange with AL language:**

1. **Creating new App:**

* Start VS Code
* Press “F1” (“Command Palette”)
* Enter “AL: Go!”
* Enter path (e.g. “…\Documents\Task2”)
* Select the latest target platform (4.0)
* In the dropdown list, pick “Your own server”
* When it asks you for credentials, don’t log in, just press ESC
* Set up the “launch.json”
* Make sure the settings are the same as the ones from your own landing page
* You can find those settings when you click “landing page” on the desktop of the VM.





Set up the “app.json”:

* Change “name” to “Task <Number of Task>”
* Change “publisher” to “<<your Name and Surname>>”
* Add feature “TranslationFile”



Download the Symbols:

* Use the Command Palette (F1) to search for “AL: Download symbols”

If necessary:

* Provide username and password from boarding pass



Delete “Hello world.al” file.

**Task 2.**

Parsing complex json response

**Sources:**

1. FIFA World CUP 2018 results API:

https://api.fifa.com/api/v1/calendar/matches?idseason=254645&idcompetition=17&language=en-GB&count=100

1. GitHub Repository with source code: github.com/abaludin/FIFA

**Step 1**

Run Request sample <https://api.fifa.com/api/v1/calendar/matches?idseason=254645&idcompetition=17&language=en-GB&count=100>

in Postman or browser. It’s official FIFA API returning detailed results of FIFA World Cup 2018 with complex structure and different data types. In Task 2 we will send request, parse json response from API, save results to table and prepare a nice view of data.

Create FIFA results table and Page (list type) with next fields:

* Match no. (Integer) \*key
* Home Team (Text)
* Away Team (Text)
* Home Team Result (Text)
* Away Team Result (Text)
* City (text)
* DateAndTime (DateTime)
* Flag (Media)

Table code:

<https://github.com/ABaludin/FIFA/blob/master/src/table/Tab50020.AB_FIFAResults.al>

Page code:

<https://github.com/ABaludin/FIFA/blob/master/src/page/Pag50020.AB_FIFAResultsList.al>

**Step 2**

Create management codeunit:

Create Refresh function:

* Make Fifa Results clearance before run (DeleteAll)
* Get response from Fifa API using universal import code from Task 1:

## procedure Refresh()

## var

## Results: Record "AB\_FIFA Results";

## URL: Text;

## Client: HttpClient;

## ResponseMessage: HttpResponseMessage;

## ResponseText: text;

## begin

## Results.DeleteAll();

## URL := 'https://api.fifa.com/api/v1/calendar/matches?idseason=254645&idcompetition=17&language=en-GB&count=100';

## if not Client.Get(URL, ResponseMessage) then

## Error(Text001\_Err);

## ResponseMessage.Content().ReadAs(ResponseText);

## if not ResponseMessage.IsSuccessStatusCode() then

## error(Text002\_Err, ResponseMessage.HttpStatusCode(), ResponseText);

## end;

Text variables:

|  |
| --- |
| Text001\_Err: Label 'Service inaccessible'; |
|  |

## Text002\_Err: Label 'The web service returned an error message:\ Status code: %1\ Description: %2';

**Step 3**

Parse response

In json response we can see json array named “Results” and Match json tokens in it with numbers from 0 to 63

So we need to create loop extracting tokens from 0 to 63 from array and inserting records based on data from tokens:

Place this code in the end of Refresh function:

## ContentToken.ReadFrom(ResponseText);

## ContentToken.AsObject().Get('Results', ResultsToken);

## for i := 0 to ResultsToken.AsArray().Count() - 1 do begin

## ResultsToken.AsArray().Get(i, MatchToken);

## InsertResults(MatchToken);

## end;

extra variables:

## ContentToken: JsonToken;

## ResultsToken: JsonToken;

## MatchToken: JsonToken;

## ResponseText: text;

## i: Integer;

Create InsertResult function accepting MatchToken: JsonToken

* Look at response text one more time and find which token compares to table’s fields. Import Flag media will be described in Step 4.
* Remember that some games ended in overtime and TeamScore may be not the final score.
* To get first level tokens use:

## MatchToken.AsObject().Get('<<Token Name>>', ValueToken);

* To get value from token use:

## ValueToken.AsValue().AsInteger();

## ValueToken.AsValue().AsText();

## ValueToken.AsValue().AsDateTime();

## E t. c.

* To get token inside token use next construction:

## ParentToken.AsObject().SelectToken('Token.SubToken', ChildToken);

Pay attention that some tokens contains array with single token inside (nobody told that it will be easy task):



It could be extracted like this:

## MatchToken.AsObject().SelectToken('Home.TeamName', ValueToken);

## ValueToken.AsArray().Get(0, ValueToken);

## ValueToken.AsObject().Get('Description', ValueToken);

## Results.HomeTeam := CopyStr(ValueToken.AsValue().AsText(), 1, MaxStrLen(Results.HomeTeam));

Code of function:

## local procedure InsertResults(MatchToken: JsonToken)

## var

## Results: Record "AB\_FIFA Results";

## ValueToken: JsonToken;

## InStr: InStream;

## begin

## Results.init();

## MatchToken.AsObject().Get('MatchNumber', ValueToken);

## Results.MatchNo := ValueToken.AsValue().AsInteger();

## MatchToken.AsObject().SelectToken('Home.TeamName', ValueToken);

## ValueToken.AsArray().Get(0, ValueToken);

## ValueToken.AsObject().Get('Description', ValueToken);

## Results.HomeTeam := CopyStr(ValueToken.AsValue().AsText(), 1, MaxStrLen(Results.HomeTeam));

## MatchToken.AsObject().SelectToken('Away.TeamName', ValueToken);

## ValueToken.AsArray().Get(0, ValueToken);

## ValueToken.AsObject().Get('Description', ValueToken);

## Results.AwayTeam := CopyStr(ValueToken.AsValue().AsText(), 1, MaxStrLen(Results.AwayTeam));

## MatchToken.AsObject().Get('HomeTeamScore', ValueToken);

## Results.HomeTeamResult := ValueToken.AsValue().AsInteger();

## MatchToken.AsObject().Get('HomeTeamPenaltyScore', ValueToken);

## If ValueToken.AsValue().AsInteger() > 0 then //were penalties in overtime - replace TeamResult

## Results.HomeTeamResult := ValueToken.AsValue().AsInteger();

## MatchToken.AsObject().Get('AwayTeamScore', ValueToken);

## Results.AwayTeamResult := ValueToken.AsValue().AsInteger();

## MatchToken.AsObject().Get('AwayTeamPenaltyScore', ValueToken);

## If ValueToken.AsValue().AsInteger() > 0 then //were penalties in overtime - replace TeamResult

## Results.AwayTeamResult := ValueToken.AsValue().AsInteger();

## if Results.HomeTeamResult - Results.AwayTeamResult <> 0 then //not import flag if draw

## If Results.HomeTeamResult - Results.AwayTeamResult > 0 then begin //Home team Win

## MatchToken.AsObject().SelectToken('Home.PictureUrl', ValueToken);

## GetFlagStream(ValueToken.AsValue().AsText(), InStr);

## Results.Flag.ImportStream(InStr, Results.HomeTeam);

## end else begin // Away team Win

## MatchToken.AsObject().SelectToken('Away.PictureUrl', ValueToken);

## GetFlagStream(ValueToken.AsValue().AsText(), InStr);

## Results.Flag.ImportStream(InStr, Results.AwayTeam);

## end;

## MatchToken.AsObject().SelectToken('Stadium.CityName', ValueToken);

## ValueToken.AsArray().Get(0, ValueToken);

## ValueToken.AsObject().Get('Description', ValueToken);

## Results.City := CopyStr(ValueToken.AsValue().AsText(), 1, MaxStrLen(Results.City));

## MatchToken.AsObject().Get('LocalDate', ValueToken);

## Results.DateAndTime := ValueToken.AsValue().AsDateTime();

## Results.Insert();

## end;

**Step 4**

Import Flag Picture

* Find picture URL in Home and Away tokens. It looks like this:

[https://api.fifa.com/api/v1/picture/flags-{format}-{size}/RUS](https://api.fifa.com/api/v1/picture/flags-%7bformat%7d-%7bsize%7d/RUS)

* Use **fwc2018** instead of {format} and **1** instead of {size}. Use new TextBuilder data type for that

<https://api.fifa.com/api/v1/picture/flags-fwc2018-1/RUS>

## TextBuilder.Append(FlagUrl);

## TextBuilder.Replace('{format}', 'fwc2018');

## TextBuilder.Replace('{size}', '1');

## TextBuilder.ToText(),

* You are able to show only one picture for record (in tiles view) so you could import Home team flag or Winner flag (a little challenge)
* Create new function to get picture stream with separate HttpClient and import it to Flag field:

## local procedure GetFlagStream(FlagUrl: Text; var InStr: InStream)

## var

## Client: HttpClient;

## ResponseMessage: HttpResponseMessage;

## ResponseText: Text;

## TextBuilder: TextBuilder;

## begin

## //Basic URL is https://api.fifa.com/api/v1/picture/flags-{format}-{size}/RUS

## TextBuilder.Append(FlagUrl);

## TextBuilder.Replace('{format}', 'fwc2018');

## TextBuilder.Replace('{size}', '1');

## if not Client.Get(TextBuilder.ToText(), ResponseMessage) then

## Error(Text001\_Err);

## if not ResponseMessage.IsSuccessStatusCode() then begin

## ResponseMessage.Content().ReadAs(ResponseText);

## error(Text002\_Err, ResponseMessage.HttpStatusCode(), ResponseText);

## end;

## ResponseMessage.Content().ReadAs(InStr);

## end;

Full code of codeunit:

<https://github.com/ABaludin/FIFA/blob/master/src/codeunit/Cod50020.AB_FIFAResultsmanagement.al>

**Step 5**

* On Fifa Results table create new Brick fieldgroup to show flag on list like this:

## fieldgroup(Brick; "Flag", MatchNo, "HomeTeam", HomeTeamResult, AwayTeam, AwayTeamResult)

Page code:

<https://github.com/ABaludin/FIFA/blob/master/src/page/Pag50020.AB_FIFAResultsList.al>

**Step 6**

Create action for Fifa Results page to call Refresh Results procedure.

## action(RefreshResults)

## {

## Caption = 'Refresh Results';

## Promoted = true;

## PromotedCategory = Process;

## Image = RefreshLines;

## ApplicationArea = All;

## trigger OnAction();

## begin

## RefreshResults();

## CurrPage.Update();

## if FindFirst() then;

## end;

## }

In my code it calls a function from Results Table, you could place them both on the page:

## procedure RefreshResults();

## var

## FifaMgt: Codeunit "AB\_FIFA Results management";

## begin

## FifaMgt.Refresh();

## end;

You need to get something like that in the end:

