

Artículo

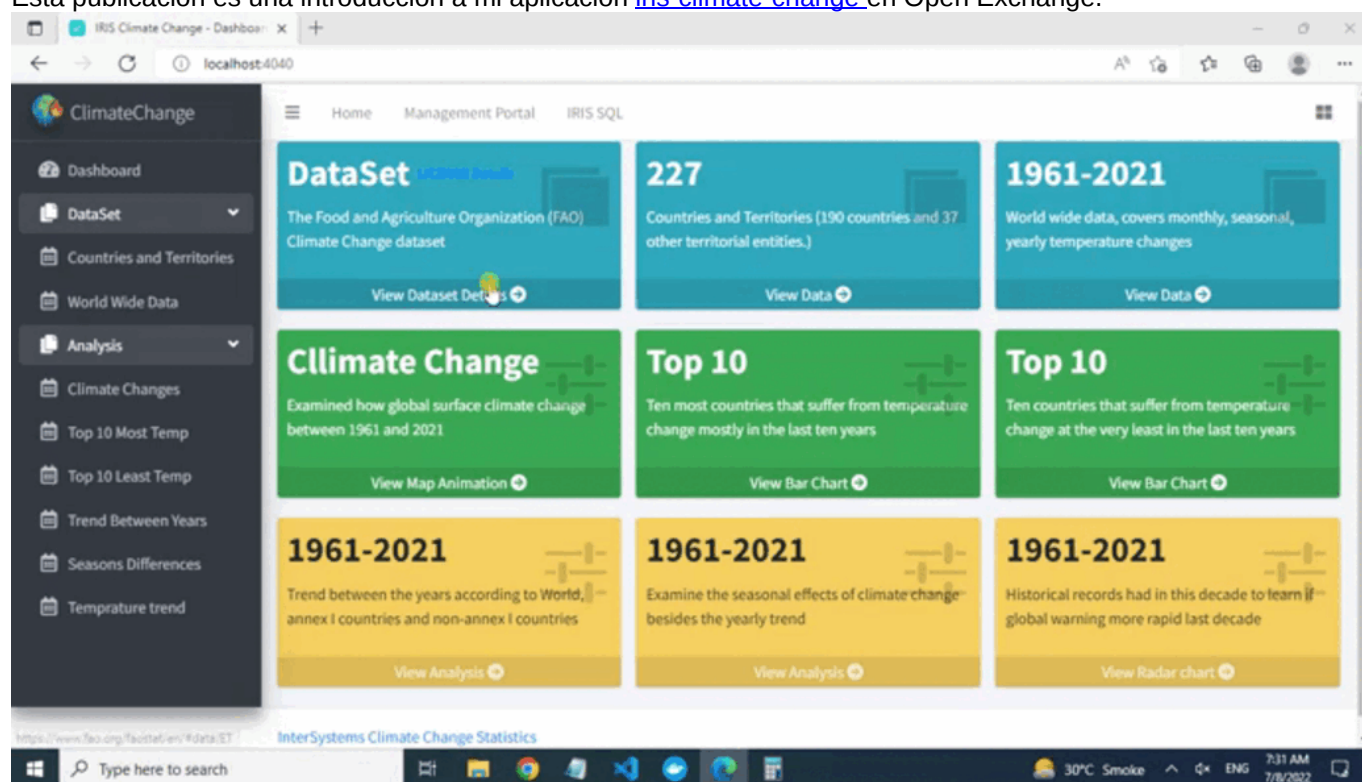
[Muhammad Waseem](#) · 12 jul, 2022 Lectura de 8 min

[Open Exchange](#)

## Aplicación IRIS Climate Change que muestra cómo la temperatura está aumentando en todo el mundo como prueba del calentamiento global

Hola Comunidad,

Esta publicación es una introducción a mi aplicación [iris-climate-change](#) en Open Exchange.



[iris-climate-change](#) importa el conjunto de datos [Temperature Change](#), de la Organización de las Naciones Unidas para la Agricultura y la Alimentación (FAO), usando la funcionalidad [LOAD DATA \(SQL\)](#) y analiza y visualiza los datos con la ayuda del framework [Python Flask Web](#), la librería [Pandas Python data analysis](#), [Plotly Open Source Graphing Library for Python](#) y la librería [Plotly JavaScript Open Source Graphing](#).

### Importación del conjunto de datos

Para importar el conjunto de datos utilizaremos el siguiente método de clase ImportDS() de la clase dc.utility:

```
Class dc.utility
{
ClassMethod ImportDS() As %String
{
    //Dynamically create country table
    Do ..GetColTypes(
```

```
"/opt/irisapp/src/data/climatechange/FAOSTAT_data_11-24-2020.csv",.coltype)
    SET tableName= "dc.ClimateChange.Countries"
    SET qry = "CREATE TABLE "_tableName_" ("_coltype_")"
    SET rset = ##class(%SQL.Statement).%ExecDirect(,qry)
    //Check if table created successfully
    IF rset.%SQLCODE
    {
        WRITE rset.%Message,!
    }
    ELSE
    {
        //Load DATA to country table
        SET qry =
"LOAD DATA FROM FILE  '/opt/irisapp/src/data/climatechange/FAOSTAT_data_11-24-2020.csv' INTO dc.ClimateChange.Countries USING {""from"":{""file"":{""header"":""0"",""skip"":""1""}}}"
        SET rset = ##class(%SQL.Statement).%ExecDirect(,qry)
        IF rset.%SQLCODE
        {
            WRITE rset.%Message,!
        }
    }

    //Dynamically create Climatechange data
    Do ..GetColTypes(
"/opt/irisapp/src/data/climatechange/Environment_Temperature_change_E_All_Data_NOFLAG.csv",.coltype)
    SET tableName= "dc.ClimateChange.Data"

//SET coltype = "AreaCode NUMERIC(3,0), Area VARCHAR(100),MonthsCode NUMERIC(4,0),Months VARCHAR(20),ElementCode VARCHAR(4), Element VARCHAR(100), UNIT VARCHAR(3), Y1961 NUMERIC(7,3)"
    SET qry = "CREATE TABLE "_tableName_" ("_coltype_")"
    SET rset = ##class(%SQL.Statement).%ExecDirect(,qry)
    //Check if table created successfully
    IF rset.%SQLCODE
    {
        WRITE rset.%Message,!
    }
    ELSE
    {
        //Load DATA statement
        SET qry =
"LOAD DATA FROM FILE  '/opt/irisapp/src/data/climatechange/Environment_Temperature_change_E_All_Data_NOFLAG.csv' INTO dc.ClimateChange.Data USING {""from"":{""file"":{""header"":""0"",""skip"":""1""}}}"
        SET rset = ##class(%SQL.Statement).%ExecDirect(,qry)
        IF rset.%SQLCODE
        {
            WRITE rset.%Message,!
        }
    }
    }
    return ""
}
```

```
// Get column lists based on the Header Row of CSV File
ClassMethod GetColTypes(filenamecsv As %String, ByRef coltype As %String, dlm As %String = ",")
```

```
{
    SET coltype=""
    SET stream=..GetStreamFromFile(filenamecsv)
    SET header=stream.ReadLine() // Get Header columns
    //SET header=$ZCONVERT(header,"L")
    SET dataLine=stream.ReadLine() // Read First line of data
    //Read all the columns of header and add datatype against it
    FOR i=1:1:($Length(header,dlm)) {
        kill types
        //Get datatype of the column
        DO ..GetData($Piece(dataLine,dlm,i),.types)
        SET type=..GetType(.types)

        // if type is varchar then assign length 250
        IF type="VARCHAR" SET type="VARCHAR(250)"
        SET $Piece(coltype,dlm,i)=$TR($Piece(header,dlm,i)," ","")_" "_type
    }
    SET coltype=$TR(coltype,dlm,"")
    // change "." symbol in column names
    SET coltype=$TR(coltype,".","_")
}
```

ClassMethod GetStreamFromFile(filename As %String) As %Stream

```
{
    // create new stream and link to the file
    SET stream = ##Class(%Stream.FileCharacter).%New()
    SET stream.LineTerminator = $Char(13,10)
    $$$TOE(sc,stream.LinkToFile(filename))
    RETURN stream
}
```

ClassMethod GetDataType(value As %String, ByRef types) As %Status

```
{
    //Get datatype based on the first column value
    if $IsValidDouble(value) {
        if $I(types("DOUBLE"))
            if $L(value,".")>1,$L($P(value,".",2))<10 if $I(types("MONEY"))
            if ($L(value,".")=1) if $I(types("INTEGER"))
        quit $$$OK
    }
    if ..IsDate(value) {
        if $I(types("DATE"))
            Quit $$$OK
    }
    if $I(types("VARCHAR"))
        return $$$OK
}
```

ClassMethod GetType(ByRef types) As %String

```
{
    If $D(types("MONEY")),$D(types("DOUBLE")) {
        if types("MONEY")=types("DOUBLE") return "MONEY"
    }
    SET i=$Order(types(""))
    while i'="" {
        SET typesI(types(i))=i
        SET i=$Order(types(i))
    }
}
```

```
    if $D(typesI) return typesI($Order(typesI(""), -1))
    return "VARCHAR"
}

ClassMethod IsDate(pVar As %String = "") As %Boolean
{
    SET sts=$$$$YES
    Try {
        If $ZDateH(pVar,5)
    }
    Catch e {
        SET sts=$$$$NO
    }
    Quit sts
}
}
```

Para mostrar los diez países que han sufrido los mayores cambios de temperatura en los últimos diez años, usaremos el siguiente código python, HTML y javascript

```
#Ten most countries that suffer from temperature change mostly in the last ten years
@app.route("/mosttemp")
def mosttemp():
    #Get DataFrame
    df = getDataFrame()
    #Create a copy of DataFrame
    df_c = df.copy()
    df_c.set_index("year", inplace=True)
    df_c = df_c.loc[['2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017',
'2018', '2019']]
    df_c.reset_index(inplace = True)
    #Group by country name
    df_c = df_c.groupby(
        ['country name',]
    ).agg(
        {
            'tem_change': 'mean',

        }
    )
    df_c.reset_index(inplace = True)
    #applying sorting
    df_c = df_c.sort_values(by=['tem_change'], ascending=False).head(10)
    #adding titles
    fig = px.bar(df_c, x="country name", y='tem_change' ,text='tem_change',
        title=
"Top ten countries that have highest temperature change in the last decades"

"<br>The top ten list shows Europe and some European countries. It also has been illu
strated that Europe"

"<br> is affected mostly by climate change. And not surprisingly, all countries on th
e list are industrialized countries,"
        "<br> excluding 'Svalbard and Jan Mayen Islands'." )
```

```
fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')

# adjusting size of graph, legend place, and background colour
fig.update_layout(
    autosize=False,
    width=1000,
    height=600,
    margin=dict(
        l=50,
        r=50,
        b=100,
        t=100,
        pad=4
    ),

    template='seaborn',
    paper_bgcolor="rgb(234, 234, 242)",
    legend=dict(
        orientation="v",
        yanchor="bottom",
        y=0.3,
        xanchor="left",
        x=1.02
    ))
fig.update_xaxes( tickangle = 10,
    title_text = "Countries",
    title_font = {"size": 15},
    title_standoff = 0)
fig.update_yaxes(showticklabels=False,tickmode="auto", title=
'Temperature Change',title_standoff = 0)
#Converting dataframe to JSON in order to display on web
graphJSON = json.dumps(fig, cls=plotly.utils.PlotlyJSONEncoder)
#rendering main.html page by passing json object
return render_template("main.html", fig=graphJSON)

def getDataFrame():
    df= pd.read_csv(
"/opt/irisapp/src/data/climatechange/Environment_Temperature_change_E_All_Data_NOFLAG
.csv", encoding='latin-1')
    #Get data from IRIS
    statement = iris.sql.exec(
'SELECT Country as "Country Name", ISO3Code as "Country Code" FROM ClimateChange.Coun
tries')
    df_countrycode = statement.dataframe()

    #Renaming columns
    df.rename(columns = {'Area':'country name'},inplace = True)
    df.set_index('Months', inplace=True)
    df.rename({'Dec-Jan-Feb': 'Winter', 'Mar-Apr-May': 'Spring', 'Jun-
Jul-Aug': 'Summer', 'Sep-Oct-Nov': 'Fall'}, axis='index',inplace = True)
    df.reset_index(inplace = True)

    #Filtering EndYear
    df = df[df['Element'] == 'Temperature change']

    #Merging with df to df_country
    df = pd.merge(df, df_countrycode, how='outer', on='country name')

    #Drop unwanted columns
```

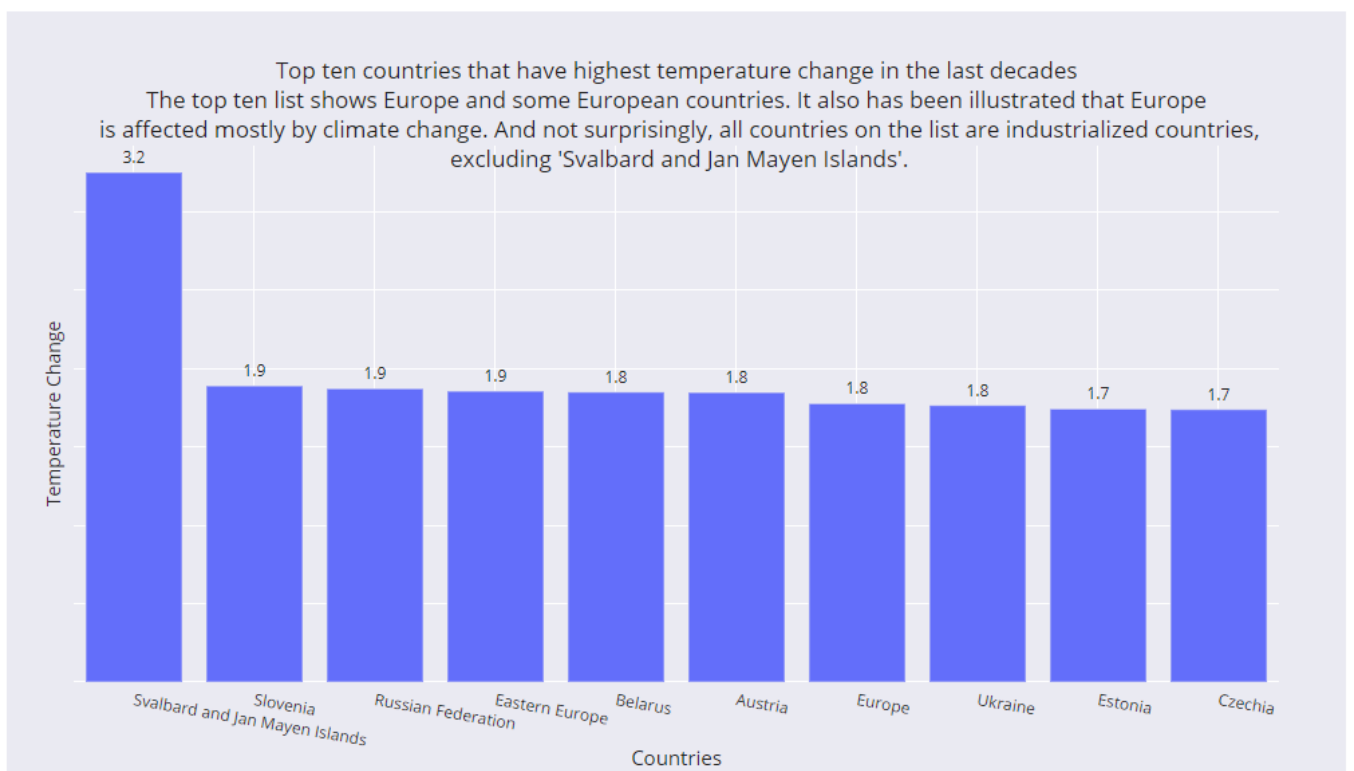
```
df.drop(['AreaCode', 'MonthsCode', 'ElementCode', 'Element', 'Unit'],axis=1
,inplace=True)

#Channing dataframe organization
df = df.melt(id_vars=["country code", "country name","Months"
,], var_name="year", value_name="tem_change")
df["year"] = [i.split("Y")[-1] for i in df.year]

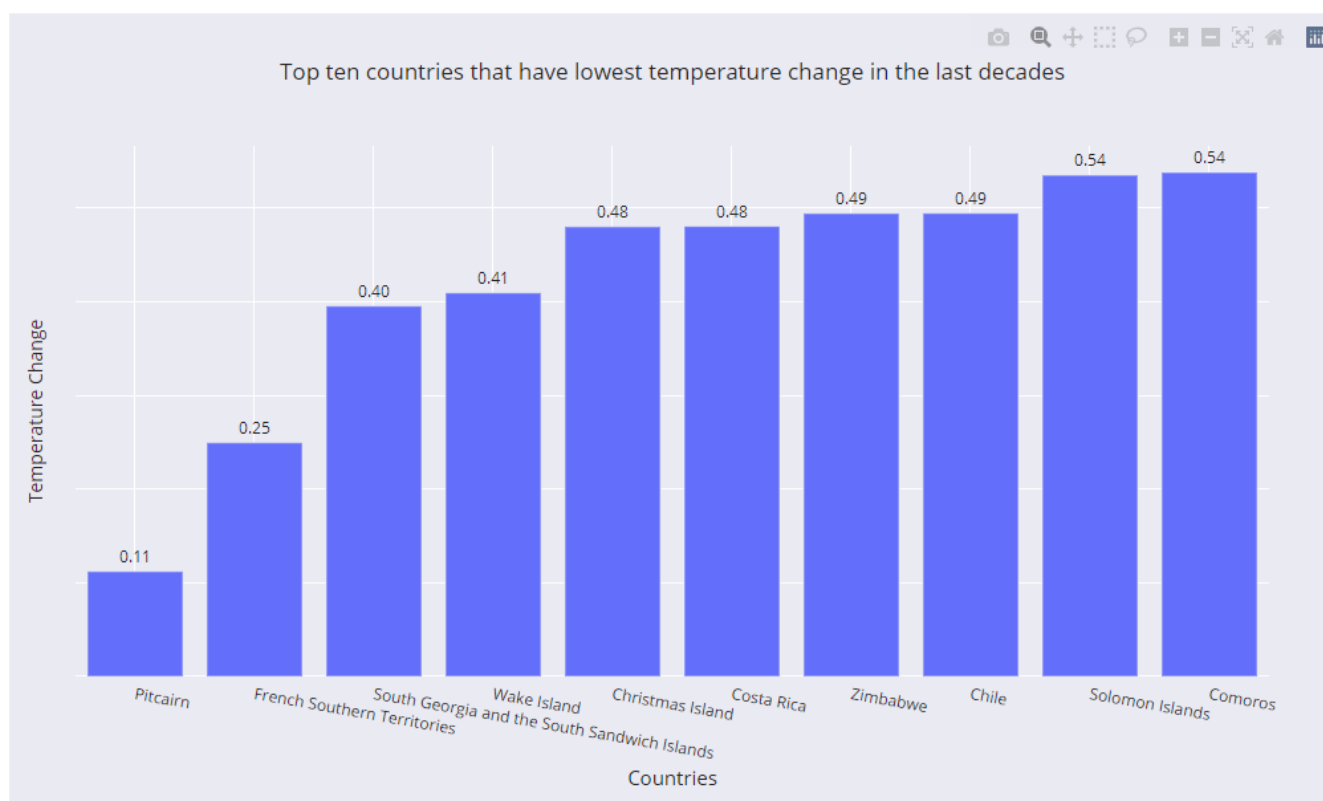
return df
```

```
<script src="/static/assets/plugins/plotly/plotly-latest.min.js"></script>
<script>
    Plotly.newPlot("myPlot", {{ fig | safe }})
</script>
```

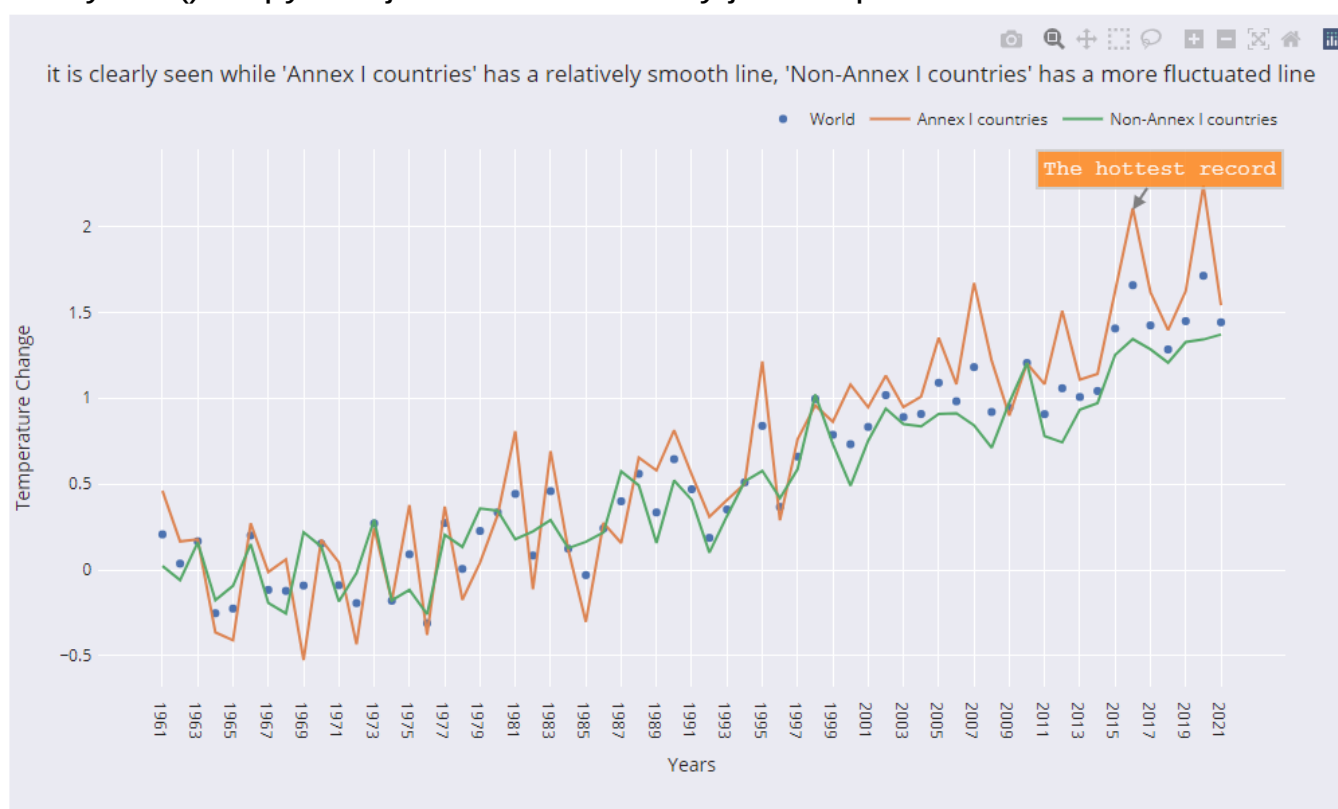
```
<div id="myPlot"></div>
```



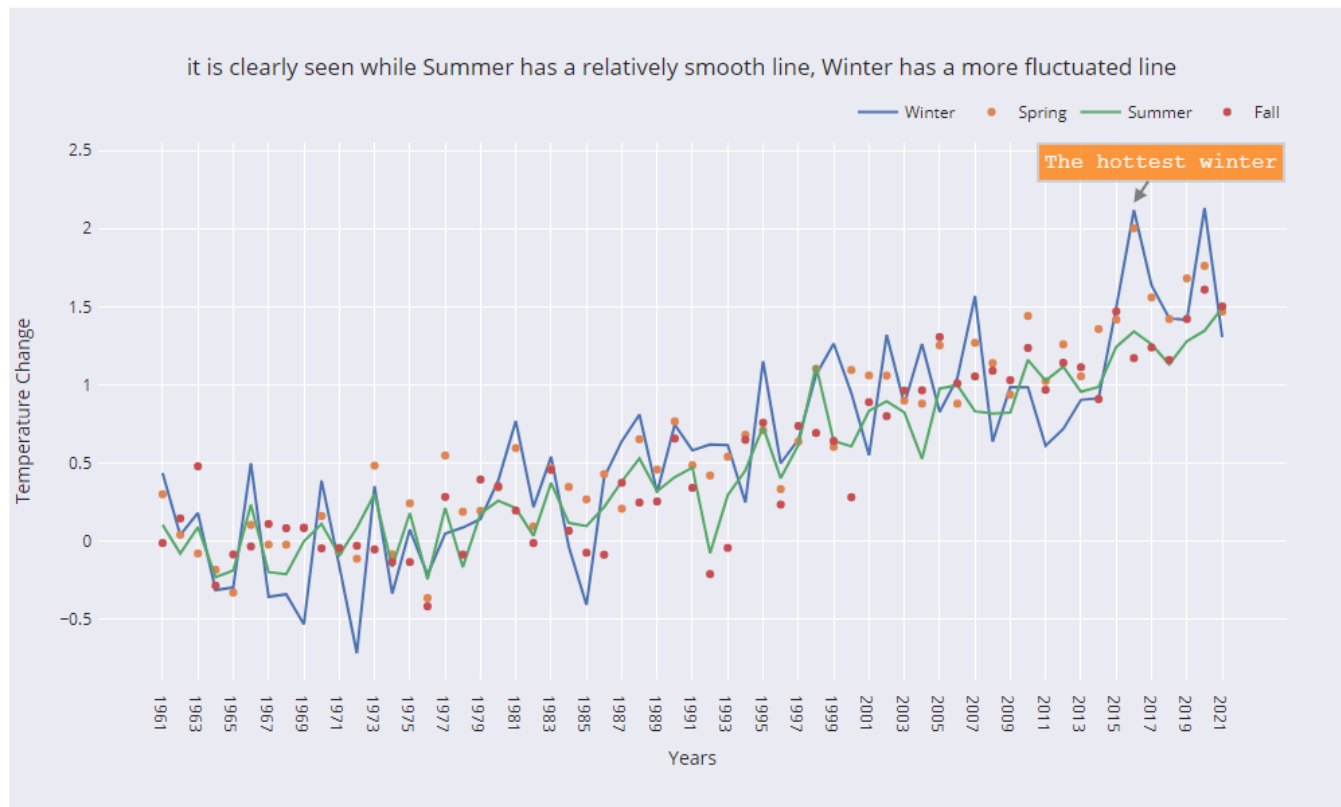
Para mostrar los diez países que han sufrido los menores cambios de temperatura en los últimos diez años, se usará la función de Python `Leasttemp()` junto con el HTML y JavaScript anteriores.



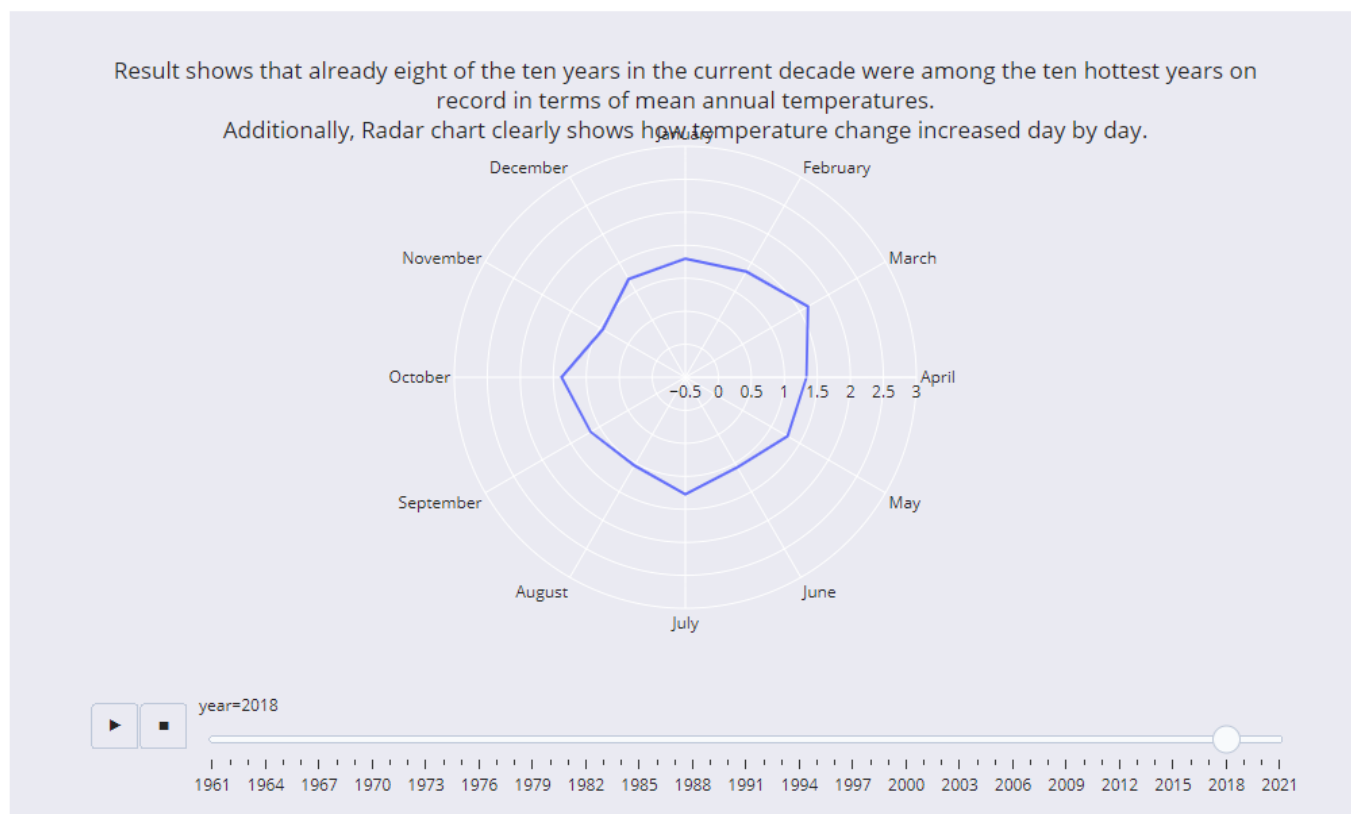
Para mostrar la tendencia a lo largo de los años en el mundo, en los países del anexo I y en los países que no pertenecen al anexo I, se usará la función `trendyears()` de python junto con el HTML y javascript anteriores.



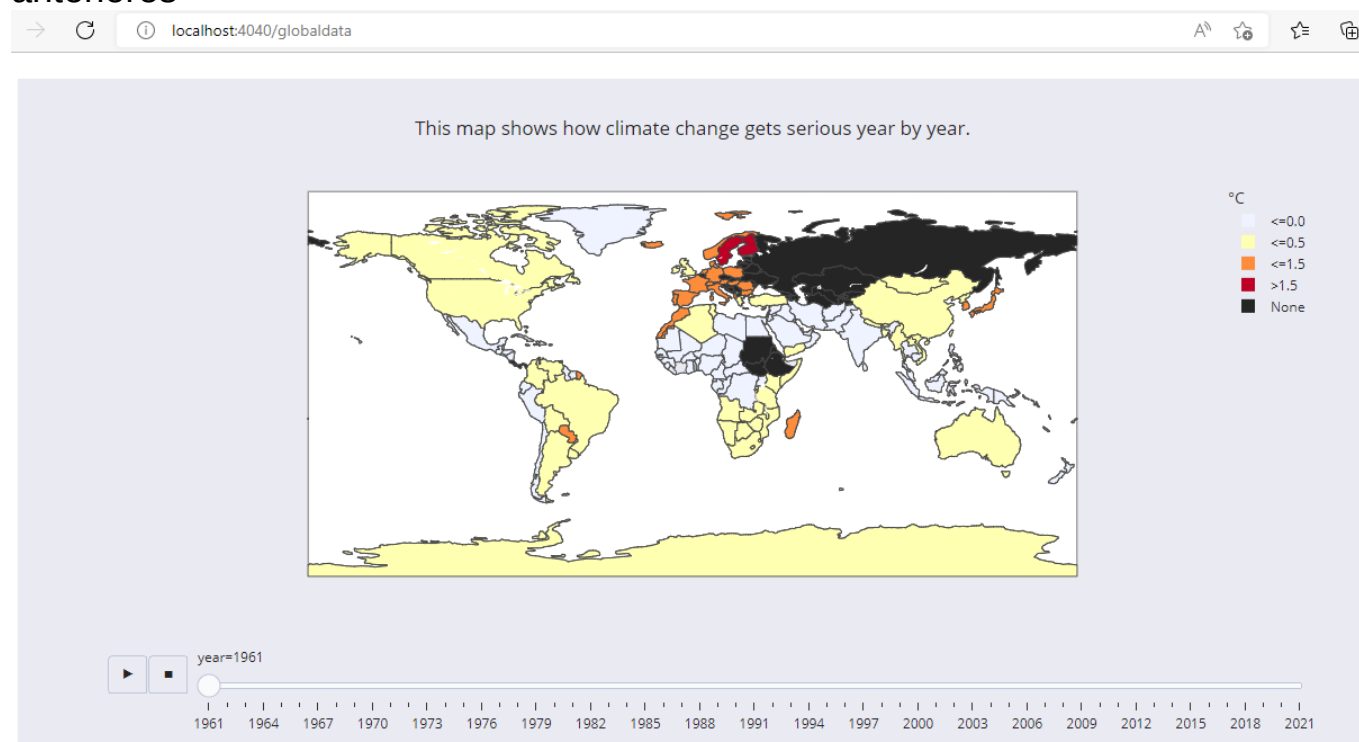
Para examinar los efectos estacionales del cambio climático además de la tendencia anual, se utilizará la función `seasons()` de Python junto con el HTML y JavaScript anteriores



Para examinar los efectos estacionales del cambio climático, además de la tendencia anual, se utilizará la función `Trendtemp()` de Python junto con el HTML y JavaScript anteriores.



Y, por último, para examinar el cambio climático global en la superficie entre 1961 y 2021 se usará la función `globaldata()` de python, junto con el HTML y javascript anteriores



## Conclusión:

Examinamos cómo cambia la temperatura de la superficie mundial entre 1961 y 2021. Al examinar las diez áreas principales que han sufrido el cambio de temperatura más alto en la última década son en su mayoría países industrializados. Además, encontramos que la temperatura aumentaba cada diez décadas, y la última década puede contarse como la década más calurosa. Del resultado de nuestro análisis anterior, nos encontramos con inviernos cada vez más calientes. Finalmente, tratamos de mostrar cómo la temperatura está aumentando en todo el mundo como prueba del calentamiento global. Analizamos los efectos más profundos del cambio climático en el mapa interactivo; esto muestra cómo el cambio climático se agrava año tras año.

Si encuentra útil esta aplicación, considérela para votar

Gracias

[#Concurso](#) [#Visualización](#) [#InterSystems](#) [IRIS](#)  
[Ir a la aplicación en InterSystems Open Exchange](#)

---

URL de  
fuente: <https://es.community.intersystems.com/post/aplicaci%C3%B3n-iris-climate-change-que-muestra-c%C3%B3mo-la-temperatura-est%C3%A1-aumentando-en-todo-el-mundo>