I was reading the DS2 documentation and ran across the HTTP package. It intrigued me. Because DS2 had no text file handling statements, I thought all hope of leveraging Internet-based APIs was lost – but apparently I despaired too soon! And where better for a SAS Jedi to hone new API wielding skills than the Start Wars API (SWAPI – find it at http://swapi.co/documentation) The site gives an API call that returns information about Luke Skywalker – http://swapi.co/people/1 and I'll use that to test my DS2 code.

Here's the DS2 program I used to do my test:

```
proc ds2 ;
data _null_;
   dcl package logger putlog();
   dcl varchar(32767) character set utf8 url response;
   method GetResponse(varchar(32767) url);
      dcl integer i rc;
      dcl package http webQuery();
      /* create a GET call to the API*/
      webQuery.createGetMethod(url);
      /* execute the GET */
      webQuery.executeMethod();
      /* retrieve the response body as a string */
      webQuery.getResponseBodyAsString(response, rc);
   end;
   method run();
      /* Construct a GET request URL to obtain vehicle ID*/
      url='http://swapi.co/api/people';
      GetResponse(url);
      put;
      putlog.log('N',URL);
      putlog.log('N',Response);
      put;
      url='http://swapi.co/api/people/1';
      GetResponse(url);
      put;
      putlog.log('N',URL);
      putlog.log('N',Response);
      put;
   end;
enddata;
run;
quit;
```

The SAS log shows what the response from the two calls to the API looked like:

```
NOTE: http://swapi.co/api/people

NOTE: {"count":82,"next":"http://swapi.co/api/people/?page=2",
"previous":null,"results":[{"name":"Luke Skywalker","height":"172"
,"mass":"77", "hair_color":"blond","skin_color":"fair",
"eye_color":"blue","birth_year":"19BBY","gender":"male",
"homeworld":"http://swapi.co/api/planets/1/",…

NOTE: http://swapi.co/api/people/1
NOTE: {"name":"Luke kywalker","height":"172","mass":"77","hair_color":"blond"
,"skin_color":"fair","eye_color":"blue","birth_year":"19BBY",
"gender":"male","homeworld":"http://swapi.co/api/planets/1/",…
```

The results are returned in JSON, and SAS doesn't yet have an engine for reading JSON, so we'll parse out the data we want using traditional text manipulation functions like SCAN and SUBSTR.

TO build our CastOfCharacters dataset, first we need to know how many entries are in the API's people database. The first URL we used contained this information early in the response string:
`{"count":82,"`

And I can extract that number using SCAN, then use it in a DO loop to execute one query for each person in the database. Inside that loop, I'll once again use SCAN to retrieve information for each character:

```
method run();
   dcl int endloop;
   url='http://swapi.co/api/people';
   /* create a GET call to the API*/
   GetResponse(url);
   endloop=scan(Response,2,'".,:{}');
   do ID =1 to endloop;
      url=cats('http://swapi.co/api/people/',id);
      /* create a GET call to the API*/
      GetResponse(url);
      do;
         name=      scan(Response,2,'".,:{}');
         height=    scan(Response,4,'".,:{}');
         mass=      scan(Response,6,'".,:{}');
         hair_color=scan(Response,8,'".,:{}');
         skin_color=scan(Response,10,'".,:{}');
         eye_color= scan(Response,12,'".,:{}');
         birth_year=scan(Response,14,'".,:{}');
         gender=    scan(Response,16,'".,:{}');
         if name ne 'Not found' then output;
      end;
   end;
end;
```

This does the job nicely, and I get the data set I've been after:

Cast of Characters

ID	name	height	mass
1	Luke Skywalker	172	77
2	C-3PO	167	75
3	R2-D2	96	32
4	Darth Vader	202	136
5	Leia Organa	150	49

You can download the complete, working code from this link.

Until next time, may the SAS be with you!
Mark