Tesco.com Grocery API - Beta 1/v1.0.0.26 Edition

REST Implementation Reference Guide

Author: Nick Lansley   Date: 9 November 2010

Summary

This reference provides a complete guide for using the Tesco.com Grocery API, Beta 1 Edition (API). This version of the API has resulted from a project created by Tesco.com's R&D Team to provide third- party programmatic access to the latest third-generation Tesco.com grocery service.

The API has two interfaces:
An XML-based interface using the Simple Object Access Protocol (SOAP), a Microsoft-developed heavyweight and elegant interface which exposes classes, properties and methods representing the entities of the grocery service (such as 'product' and 'basket').

A simpler HTTP-based interface using elements of the Representation State Transfer (REST) methodology. This interface uses commands and parameters embedded in HTTP URL requests, known as query-strings. The interface responds with data in a format called JavaScript Object Notation (JSON).

This reference guide concentrates solely on the REST interface. Since the REST interface is implemented in a similar way to the way a web browser and web server communicate, you can learn and test the REST interface using a standard web browser, once you have obtained your developer and application keys from the Tesco API developer portal.

Introduction

This reference guide details the operation of the REST implementation complete with examples. In order to use the API you need to register for a developer key from our developer portal website at http://www.techfortesco.com/tescoapiweb. Registration is immediate and free, and a key is generated automatically for you after registration completes. The developer key is a 20-character randomly generated alphanumeric string of characters which you supply to the API when logging in a customer.

In addition to the developer key, you use the developer portal to generate one or more application keys reflecting the applications you wish to create that uses the API. This is a second 20-character randomly generated alphanumeric string of characters which you also supply to the API when logging in a customer. Most developers find that they can do all their development work with one application key - but the choice is yours.

Once you have your developer key and application key, you are ready to use the API immediately, both with a web browser (to try it out) and your applications.

Please note that the API's RESTful interface is not a pure implementation of the RESTful methodology; it only uses the HTTP GET method and query strings for all commands. This implementation is easiest both to understand and develop for, and works with more mobile software development kits and on just about every internet-aware device! However it is not a pure nor particularly elegant implementation of REST. *That will come later!*

Changes since previous edition of API and Documentation (since 1.0.0.4) consists of a couple of new commands (highlighted in **blue** text in this document), and a couple of new parameters to existing commands (also highlighted in **blue** text) alongside numerous behind-the-scenes bug fixes and performance improvements.

The API can be used by anyone subject to the terms and conditions posted at:
http://www.techfortesco.com/tescoapiweb/terms.htm

The API Syntax

The syntax for calling the RESTful API consists of:

1. The standard HTTP prefix and domain name/path to the REST endpoint.
2. A set of required and optional parameters as 'name=value' pairs separated by the "&" ampersand symbol:
    i. The parameter name can be upper or lower-case (or a mix of case).
    ii. The parameter value may need to be in the right case, especially if it is a piece of data that the API will save (such as 'NotesForShopper=' parameter in the ChangeBasket command). If the value has to contains characters that are not allowed in an HTTP path, the value will have to be "URL-encoded". This mainly involves space characters which have to be replaced by either a "+" symbol or "%20" hex-character representation (space is hex character 20). More details in those commands that require it.

Example:
`http://www.techfortesco.com/groceryapi/RESTService.aspx?COMMAND=LISTFAVOURITES&PAGE=1&SESSIONKEY=h38dhj348dfhj348dfhj48drhj4834hj4894jh494j494j484y`
Endpoint name/path: http://www.techfortesco.com/groceryapi/RESTService.aspx
Parameter: COMMAND=LISTFAVOURITES
Parameter: PAGE=1
Parameter: SESSIONKEY=h38dhj348dfhj348dfhj48drhj4834hj4894jh494j494j484y
Additional syntax characters:
"?" between endpoint name/path and first parameter
"=" between parameter name and its value
"&" between parameters.

Understanding responses by the API

The REST implementation of the API responds to commands using the JavaScript Object Notation (JSON) format. JSON is a lightweight data format compatible with the JavaScript programming language. For more information on the JSON format see http://www.json.org .

The API implements JSON responses with header data and, if applicable, an array list of repeated items - for example products, delivery slots or pending orders.

The API also implements the JSONP extension, which wraps the JSON response between curved brackets preceded by the name of the JSONP value. JSONP is used in dynamic web pages that uses the JSONP 'effect' to cause the browser to automatically call a function that results from this API call. Here are two examples:
Without JSONP parameter
`http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=READYFORCHECKOUT&sessionkey=2tts7VPx3aVkOK1LCL1WydA58JoAixxj8DZK5msFuClEt5eXdo`
Returns:

```
{ "StatusCode": 0, "StatusInfo": "9 items in basket, and delivery slot reserved for delivery between 2010-05-18 17:00 and 19:00 - Checkout OK to proceed.", "ItemsInBasket": 9, "DeliverySlotStart": "2010-05-18 17:00", "DeliverySlotEnd": "2010-05-18 19:00", "DeliverySlotReservationExpires": "2010-05-13 19:05", "ReadyForCheckout": "Y" }
```

With a JSONP parameter that requests the JSON to be wrapped in a function called 'MyFunction':
`http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=READYFORCHECKOUT&`**`JSONP=MyFunction`**`&sessionkey=2tts7VPx3aVkOK1LCL1WydA58JoAixxj8DZK5msFuClEt5eXdo`
Returns:

```
MyFunction( { "StatusCode": 0, "StatusInfo": "9 items in basket, and delivery slot reserved for delivery between 2010-05-18 17:00 and 19:00 - Checkout OK to proceed.", "ItemsInBasket": 9, "DeliverySlotStart": "2010-05-18 17:00", "DeliverySlotEnd": "2010-05-18 19:00", "DeliverySlotReservationExpires": "2010-05-13 19:05", "ReadyForCheckout": "Y" })
```

Status Codes

Every JSON-formatted response contains the values StatusCode and StatusInfo, which can be examined to assess the success or otherwise of your command.
`"StatusCode": 0, "StatusInfo": "Command Processed OK",`

StatusCode 0: The API believes it did a splendid job processing your request successfully and encountered no problems.

100-series: User/client app input errors / or input that is no longer valid. Usage: The API is not accepting your input. The StatusInfo will explain what's wrong.

StatusCode: 100 StatusInfo: "NO Delivery Slot reserved"
StatusCode: 105 StatusInfo: "Fewer than 5 items in the basket" (used with new 'ReadyForCheckoutTest' function)
StatusCode: 110 StatusInfo: "Invalid input for Rating (must be 1,2,3,4 or 5)"
StatusCode: 115 StatusInfo: "Invalid or missing session key"
StatusCode: 120 StatusInfo: "Lost or corrupted session - please login again and get a new session key"
StatusCode: 125 StatusInfo: "Sorry your requested delivery slot could not be reserved - it may have become full in the time between receiving the slot-list and choosing the slot."
StatusCode: 130 StatusInfo: "No Delivery slots returned - session timeout? Try LOGIN again."
StatusCode: 135 StatusInfo: "This category is no longer available or is not a shelf (it has one or more children)"

200-series: Anonymous mode.
Usage: Some commands are not available with API sessions running in anonymous mode (this mode is where email and password are not supplied to the Login command).
How to react: Your application should be hiding such functions as "add to basket" until the customer supplies their login credentials. Disable any command that tells the API to store information (for example, add to basket) or to access personal account ifno such as favourites. Anonymous mode is giving you search access only.
StatusCode: 200 StatusInfo: "This command not available in Anonymous mode", StatusInfo: "Sorry, branch <n> does not have an anonymous login facility", StatusInfo: "Sorry, changing anonymous branch requires anonymous login (use Login command / method with email and password set to empty strings).".

600-series: Developer authentication issues
Usage: The terms and conditions for use of the API allows us to suspend or cancel developer / application keys (or limit API usage per day) because of unusual or nefarious use.
How to react: Apologise to the user and tell them you will be in touch with Tesco.com to deal with the issue as soon as possible.
StatusCode: 600 StatusInfo: "Developer Key has used up today's API access call count", <= used in rare circumstances StatusInfo: Developer Key and/or Application Key not found or disabled".

900-series: API / Grocery internal system errors. Usage: If the API encounters problems accessing the Tesco grocery service, whether communication, response times or errors parsing returned pages, it returns error
900. It will also describe the internal error which will be useful to use when debugging the problem.
How to react: Tell the customer that an error was encountered and ask them to try that function again later.
StatusCode: 900 StatusInfo: "(Internally generated system error message)"

Product Pagination

Product searches can yield a large number of results given that the average customer's available products are 25,000! To overcome this very real problem (especially for mobile devices), the API implements 'pagination' - that is, the results are broken into groups called 'pages' and only a single group or 'page' is returned.

Product Pagination is implemented in commands PRODUCTSEARCH, LISTFAVOURITES and LISTPRODUCTOFFERS.
When you use these commands, use the optional parameter "PAGE=1" which requests the API to return only the first page of results. In the 'header' information you will see this data:

```
"StatusCode": 0, "StatusInfo": "Command Processed OK", "PageNumber": 1,
"TotalPageCount": 12, "TotalProductCount": 237, "PageProductCount": 20,
"Products": (...list of products...)
```

A good practice is always to include "PAGE=1" when you first send the request, then use the returned information to re-request the product list with PAGE=2, PAGE=3, and so on until you you reach TotalPageCount - total number of pages for this set of results. You could also use a progress bar to show how fast your application is retrieving the results.
Note that "PAGE=0" always returns all products in a single response (even if its 1000 products or more) - it's the equivalent of not supplying the "PAGE=" parameter at all.

Product Searching and Extended Information

It is easy for the API to supply all kinds of extended information about products, including Recommended Daily Allowance (RDA), ingredients list and nutritional information.
However it is important to request the API "wisely" for this data, because the amount of data returned can be large and the time required for the API to gather the information, format and return it to your client application can be lengthy.

Here is the best practice for using this feature:

1. When your customer searches for products (whether by text search, barcode, listing favourites or offers), avoid using the ExtendedInfo parameter or set EXTENDEDINFO=N in the SEARCHTEXT command in order to get back basic information (which is quite considerable in itself, and includes links to images and any offer info where applicable).
2. Provide a facility for the customer to highlight a product, whereupon you supply the 9-digit product ID to the SEARCHTEXT parameter of the PRODUCTSEARCH command (see example above) and use EXTENDEDINFO=Y to return the extended information for that one product (see example JSON on next page).

On the next page you will see highlighted elements that you will find useful for your customers to see, although you must bear in mind that this data can vary wildly between products. On some products we don't include all extended info. For example on some sweet and chocolate products there is no RDA information because quite frankly you would be treating yourself and you don't want to spoil the moment by learning how deliciously bad the product is for you...

What about searching for product that are "free-from" an ingredient or below a certain number of grammes of fat per portion?

Currently the API is unable to provide the filtering itself so in these circumstances it may well be necessary for you to request extended info for a product search. We accept that this is a useful feature and, going forward we aim the API to be able to do this itself. However only a production API with access to the product database will be able to achieve this. Until then one way of achieving best practice is to have your own server that uses our API and also processes requests form your client applications. You would design a service that would queue requests from the client application, perform the extended search, filter out those products not satisfying the nutrition or ingredients request, and then alert that application that the data was ready for it.

We have a continuous ongoing programme of improvements to product information so expect this data to improve over time as well as give the API fast filtering access to the product database which it does not have today for security reasons.

Understanding Amend Order Mode

As well as enabling your customer to build a basket of products for new orders, the API also provides the facility to amend an order that they have already placed and is pending delivery.

Use the LISTPENDINGORDERS command to list any such pending orders, and ask the customer to choose one of these orders. Take the order number linked to the selected order and use the AMENDORDER command. This causes the current basket to be ‘hidden’ (but not lost) and replaces it with the contents of the basket of the pending order. The customer’s account is now in ‘Amend Order Mode’.

Use the CHANGEBASKET command to add / amend / remove products in the basket as normal. Product searches will work as normal, too.

Finally use the SAVEAMENDORDER command to save these changes. After saving changes, SAVEAMENDORDER then returns the customer’s account to “Normal Mode” and the basket hidden by Amend Order Mode is restored. Note that when you execute the SAVEAMENDORDER command, the grocery service will email the customer immediately with details of the amended order. However, sometimes the API cannot save the amended order because the customer may need to re-authorise their payment card. This happens if the value of the amended order changes significantly as a result of the amendment. When this happens: SAVEAMENDORDER fails, the customer's account remains in Amend Order Mode and you must tell the customer to confirm the changes using the web site.

Alternatively use the CANCELAMENDORDER command which abandons any changes to the basket and restores ‘Normal Mode’. The order being amended is not lost, canceled or damaged in any way. CANCELAMENDORDER does not cause an email to be sent to the customer.
You can check if an account is in Amend Order Mode by executing the LISTBASKET or LISTBASKETSUMMARY command, and noting the following values for “BasketID” and “InAmendOrderMode” in the JSON response:

```
"BasketID": "7283288",
"InAmendOrderMode": "Y"
```

The BasketID uniquely and consistently identifies the current basket (in either mode), which you may find useful to store alongside basket contents held in your application. The InAmendOrderMode value can be “Y” if the account is in Amend Order Mode, or “N” if in ‘Normal’ mode.

Important points to note:

1. When you switch a customer’s account into Amend Order Mode, it remains in that mode until you explicitly cancel or save the amended order. Indeed, the grocery service will only force off Amend Order Mode on the eve of the delivery of that order - and it does so by cancelling changes.
2. Beware of your application misunderstanding which mode a customer’s account is in. For example, if you have implemented offline/online synchronisation, your app may synchronise with the wrong order, causing products from an offline session to flood into the wrong online basket. To avoid this, use the BasketID value available to you in LISTBASKET and LISTBASKETSUMMARY commands and save this with any products you save in an offline basket. Before you sync, use LISTBASKETSUMMARY to check which mode you are in and BasketID you are using. If the customer logs in to their Tesco grocery account on the web or through another app, they will see that their account is in Amend Order mode. They might save changes or cancel out of Amend Order Mode, so always check the current mode from time to time in a very long session involving your application.
3. Not all API functionality is available in Amend Order Mode. For example, you find that LISTPENDINGORDERS always returns an empty response. You must always cancel or save out of Amend Order Mode to restore full functionality.
4. Always aim to switch a customer account out of Amend Order Mode at the end of a session if your application switched it into Amend Order Mode in the first place.
5. If your application encounters Amend Order Mode unexpectedly, warn the customer and offer them the option to save changes (using SAVEAMENDORDER) before adding products to their (Normal Mode) basket.
6. In our risk analysis, we are fully aware of the possibility of a viral application that phishes for customer login details and then adds unwanted products to amended orders (which the customer may not notice). Any such nefarious activity will be detected by daemon processes that monitor the API service, and the developer and application keys blocked. *Don’t use the API maliciously!*

API Endpoints and Secure Access

The API has three endpoints:
http://www.techfortesco.com/groceryapi_b1/restservice.aspx - stable beta 1 endpoint
http://www.techfortesco.com/groceryapi/restservice.aspx - stable nightly build
http://www.techfortesco.com/groceryapi_ops/restservice.aspx - unstable ops-test build

You should aim to use the Beta 1 endpoint in most circumstances. However any quick bug fixes / performance improvements are applied to the Nightly Build endpoint first so this latter endpoint will always be the most up to date. Only when we are convinced that changes are proven to be successful do we apply them to the Beta 1 endpoint.
The operations-test endpoint is for use as we test updates with our existing suite of apps. We might be fixing 'live' so be prepared for a roller-coaster ride if your apps use this endpoint!

Note that all API commands can be used in secure (https) mode for full security. The LOGIN command *must* use secure mode. Just change the domain name part of the endpoint URL thus:
http://www.techfortesco.com - standard (insecure) domain name.
https://secure.techfortesco.com - SSL secure domain name.

In SSL secure mode, the client and server only exchange the domain name in clear text in order to set up the encrypted 'channel'. The rest of the path (e.g. "/groceryapi_b1/restservice.aspx") and all the parameters you supply in the rest of the API call are sent inside the encrypted channel and cannot be read by outsiders.

Command Reference Summary

COMMAND	Purpose
AMENDORDER	Switches the API into ‘Amend Order’ Mode
CANCELAMENDORDER	Cancels any edits to the amended order and returns to the current un-checked-out basket.
CHANGEBASKET	Adds products to the basket, removes them from the basket, and updates the basket.
CHOOSEDELIVERYSLOT	Reserves a delivery slot chosen from the list of delivery slots provided by ListDeliverySlots
LATESTAPPVERSION	Returns your app’s latest version (set by you in the developer portal).
LISTBASKET	Lists the contents of the basket - now with new fast responding parameter ‘FAST=Y’
LISTBASKETSUMMARY	Lists just summary information about the basket
LISTDELIVERYSLOTS	Lists available delivery slots
LISTFAVOURITES	Returns the products in the customer’s favourites list
LISTPENDINGORDERS	Lists orders that have already been checked-out but not yet delivered (these can be amended in a future update to the API).
LISTPRODUCTCATEGORIES	Lists the departments, aisles and shelves in a nested format.
LISTPRODUCTOFFERS	Lists all the products currently on offer
LISTPRODUCTSBYCATEGORY	Lists the products for a given shelf (provided by ListProductCategories. Now with new parameter ‘EXTENDEDINFO=Y’
LOGIN	Logs the customer into their Tesco.com grocery account and, if successful, returns a 20-character alphanumeric session key for use with other commands.
PRODUCTSEARCH	Searches for products using text or barcode
READYFORCHECKOUT	Checks to see if an order is ready for checkout (that is, there are at least 5 products in the basket and a delivery slot has been selected).
SAVEAMENDORDER	The API is requested to save changes to the amended order.
SERVERDATETIME	Returns the server’s current date and time.

AMEND ORDER

Purpose: Takes an order number supplied by the LISTPENDINGORDERS command and makes it the current basket to be edited. This command returns the basket lines of the order that is now being edited.

Example:
http://www.techfortesco.com/groceryapi_b1/restservice.aspx?
command=AMENDORDER&ORDERNUMBER=1234567&sessionkey=JYbMVtiaIPuGR
WiHuPQxcSAGUKDpHoyw5l3U7L5fXF4c2Khlq2

Example returned JSON:

```
{ "StatusCode": 0, "StatusInfo": "", "OrderNumber": "1234567",
"BasketGuideMultiBuySavings": "1.3", "BasketGuidePrice": "55.1",
"BasketQuantity": "29", "BasketTotalClubcardPoints": "110", "BasketLines":
[ {
"BasketLineErrorMessage": "", "BasketLineGuidePrice": "27.4",
"BasketLinePromoMessage": "", "BasketLineQuantity": "10", "BaseProductId":
"", "EANBarcode": "5011835101379", "ImagePath": "http://img.tesco.com/
Groceries/pi/379/5011835101379/
IDShot_60x60.jpg", "MaximumPurchaseQuantity": 99, "Name": "Green And Blacks
Organic Hot Chocolate 300G", "OfferPromotion": "", "OfferValidity": "",
"Price": 2.74, "PriceDescription": " (£0.91/100g)", "ProductId": "257477312",
"ProductType": "", "StorageInfo": "", "UnitPrice": 0, "UnitType": ""
}, ... (next basket line, etc) ] }
```

Parameter Name	Parameter Value
COMMAND	AMENDORDER
ORDERNUMBER	Order number from LISTPENDINGORDERS command
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

CANCEL AMEND ORDER
Purpose: Abandons changes to the basket of an order being amended, and returns the session to ‘normal’ mode, restoring the contents of the basket that were there before the account was switched into Amend Order Mode.
Example: http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=CANCELAMENDORDER&sessionkey=JYbMVtiaIPuGRWiHuPQxcSAGUKDpHoyw5l3U7L5fXF4c2Khlq2
Example returned JSON: `{ "StatusCode": 0, "StatusInfo": "Amend Order cancelled successfully" }`

Parameter Name	Parameter Value
COMMAND	CANCELAMENDORDER
SESSIONKEY	50-character session key
JSONP	Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

CHANGE BASKET
Purpose: Enables products to be added to, removed from, and updated in the current basket.
Example: http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=CHANGEBASKET&PRODUCTID=265612204&CHANGEQUANTITY=1&SUBSTITUTION=Yes&NOTEFORSHOPPER=green+ones+please&sessionkey=JYbMVtiaIPuGRWiHuPQxcSAGUKDpHoyw5l3U7L5fXF4c2Khlq2

Example returned JSON:
`{`
`"StatusCode": 0,`
`"StatusInfo": "Basket change completed successfully"`
`}`

Parameter Name	Parameter Value
COMMAND	CHANGEBASKET
SESSIONKEY	50-character session key
PRODUCTID	9-digit Product ID available in product data returned from search commands PRODUCTSEARCH, LISTFAVOURITES, LISTPRODUCTSBYCATEGORY, LISTPRODUCTOFFERS and LISTBASKET.
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.
CHANGEQUANTITY	A positive or negative value that changes the products in the basket by that quantity, according to these rules: 1) If the product was absent from the basket before that product was added, it is inserted into the basket at the requested quantity. 2) If the product was already in the basket, the quantity is increased by requested quantity if positive, or reduced by the requested quantity if the requested quantity is negative. 3) If a negative requested quantity is equal to or larger than the existing quantity, the product is removed from the basket. 4) For products that sell by weight, quantities added or removed are still "each". For example, if you are adding apples that are priced per Kg, selecting "2" for this parameter will add 2 individual apples to the basket, not 2 Kg of apples.

Parameter Name	Parameter Value
SUBSTITUTION	The allowed values are (without quotes): 1) “YES” (substitute with anything reasonable) 2) “NO” (do not substitute)
NOTEFORSHOPPER	A short description to help the shopper choose something appropriate. Try to keep this below 50 characters or an incomplete sentence may appear on the personal shopper’s screen when the product is being picked. If using characters that cannot be sent in an HTTP web address, you will need to “URLEncode” them. For example, spaces need to be converted either into “+” symbols or the %20 hex symbol. Most software development kits and computer languages support a URLEncode() method or function.

CHOOSE DELIVERY SLOT
Purpose: Selects a delivery slot from a list provided by LISTDELIVERYSLOTS
Example: http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=CHOOSEDELIVERYSLOT&DELIVERYSLOTID=d-213120100520090000201005201100&sessionkey=JYbMVtiaIPuGRWiHuPQxcSAGUKDpHoyw5l3U7L5fXF4c2Khlq2

Example returned JSON
Success example:
`{`
`“StatusCode": 0,`
`“StatusInfo": "Your delivery slot is reserved for 02:00 hours until 17:16",`
`"ReservedUntil": "2010-05-13 17:15"`
`}`
 	
Fail example:
`{`
`"StatusCode": 125,`
`"StatusInfo": "Sorry your requested delivery slot could not be reserved - it may have become full in the time between receiving the slot-list and choosing the slot.",`
`"ReservedUntil": "2000-01-01 00:00"`
`}`

Parameter Name	Parameter Value
COMMAND	CHOOSEDELIVERYSLOT
SESSIONKEY	50-character session key
DELIVERYSLOTID	One of the delivery slot IDs returned by LISTDELIVERYSLOTS
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

LATEST APP VERSION

Purpose: returns the version number for the latest version of the application and a web page that can be displayed for more info (both set in the developers portal web site) so that your app can proactively check and request that the customer gets the newer version if necessary. Note: You do not have to have executed the LOGIN command before you use this command.

N.B. The developer portal will be updated to allow you to enter the data required against each of your application keys by end November 2010. For each application key you will be able to enter a version number (LatestAppVersion) and information web address (InfoURL) which will appear in the response below.

Example:
http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=LATESTAPPVERSION&appkey=12345667890123455666554

Example returned JSON:
```
{
"StatusCode": 0,
"StatusInfo": "Command Processed OK",
"LatestAppVersion": 1.01,
“InfoURL”: “http://www.tesco.com/apps”
}
```

Parameter Name	Parameter Value
COMMAND	LATESTAPPVERSION
APPKEY	20-character application key set up in the developer’s portal
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

LIST DELIVERY SLOTS
Purpose: Lists delivery slots available, so one can be reserved with CHOOSEDELIVERYSLOT. Note - Delivery Slot Ids can be of variable length up to 100 characters and all characters have been URLEncoded by the API so they can be used with the CHOOSE DELIVERY SLOT function without any further formatting.
Example: http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=LISTDELIVERYSLOTS&sessionkey=JYbMVtiaIPuGRWiHuPQxcSAGUKDpHoyw5l3U7L5fXF4c2Khlq2

Example returned JSON:
`{`
`"StatusCode": 0,`
`"StatusInfo": "Command Processed OK",`
`"DeliverySlots":`
` [`
`   {`
`    "DeliverySlotId": "d-213120100520090000201005201100",`
`    "BranchNumber": "2131",`
`    "SlotDateTimeStart": "2010-05-20 09:00",`
`    "SlotDateTimeEnd": "2010-05-20 11:00",`
`    "ServiceCharge": 5.00`
`   },`
`   { .... more slots .....`
`   }`
` ]`
`}`

Parameter Name	Parameter Value
COMMAND	LISTDELIVERYSLOTS
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

LIST BASKET

Purpose: Lists the contents of the customer’s basket

Example:
http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=LISTBASKET&FAST=N&sessionkey=JYbMVtiaIPuGRWiHuPQxcSAGUKDpHoyw5l3U7L5fXF4c2Khlq2

Example returned JSON:

```
{
"StatusCode": 0,
"StatusInfo": "Command Processed OK",
"InAmendOrderMode": "N",
"BasketGuideMultiBuySavings": "0",
"BasketGuidePrice": "6.92",
"BasketQuantity": "4",
"BasketTotalClubcardPoints": "22",
"BasketLines":
[
{
"BasketLineErrorMessage": "",
"BasketLineGuidePrice": "3.05",
"BasketLinePromoMessage": "",
"BasketLineQuantity": "1",
"BaseProductId": "",
"EANBarcode": "5035500003003",
"ImagePath": "http://img.tesco.com/Groceries/pi/003/5035500003003/
IDShot_60x60.jpg",
"MaximumPurchaseQuantity": 99,
"Name": "Hadleigh Maid Acticoa Dark Chocolate Cranberries",
"OfferPromotion": "",
"OfferValidity": "",
"Price": 3.05,
"PriceDescription": " (£2.18/100g)",
"ProductId": "264836426",
"ProductType": "",
"StorageInfo": "",
"UnitPrice": 0,
"UnitType": "",
"NoteForPersonalShopper": "",
"SubstitutionNote": ""
},
{ .... next product ...
}
]
}
```

Parameter Name	Parameter Value
COMMAND	LISTBASKET
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

Parameter Name	Parameter Value
FAST=Y or N	(Optional) massively speeds up retrieval of the basket at the cost of not being able to find all of the core attributes required for a product, such as EANBarcode. Use FAST=Y to get the API to abandon further searching to retrieve all the core attributes when retrieving the basket. Test this mode to see if your application can cope without the missing attributes.

LIST BASKET SUMMARY
Purpose: Lists the contents of the customer's basket in summary form - ideal for use, for example, when comparing online and offline baskets on a mobile application in order to synchronise any differences.
Example: http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=LISTBASKETSUMMARY&sessionkey=JYbMVtiaIPuGRWiHuPQxcSAGUKDpHoyw5l3U7L5fXF4c2Khlq2
Example returned JSON for a basket containing two items:

```
{
"StatusCode": 0,
"StatusInfo": "Command Processed OK",
"BasketLines":
[
{
"BasketLineQuantity": "1",
"ProductId": "264836426"
},
{
"BasketLineQuantity": "3",
"ProductId": "265612204"
}
]
}
```

Parameter Name	Parameter Value
COMMAND	LISTBASKETSUMMARY
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.
INCLUDEPRODUCTS	(Optional) set to INCLUDEPRODUCTS=N if you only wish to retrieve the basket header information. Default is INCLUDEPRODUCTS=Y

LIST FAVOURITES

Purpose: Lists the products in a customer’s favourites list. Page pagination is available for this command.

Example:
http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=LISTFAVOURITES&page=1&sessionkey=JYbMVtiaIPuGRWiHuPQxcSAGUKDpHoyw5l3U7L5fXF4c2Khlq2

Example returned JSON:

```
{

"StatusCode": 0,
"StatusInfo": "Command Processed OK",
"PageNumber": 1,
"TotalPageCount": 3,
"TotalProductCount": 205,
"PageProductCount": 100,
"Products":
[
{
"BaseProductId": "50825584",
"EANBarcode": "0266800000000",
"CheaperAlternativeProductId": "",
"HealthierAlternativeProductId": "",
"ImagePath": "http://img.tesco.com/Groceries/pi/000/0266800000000/
IDShot_90x90.jpg",
"MaximumPurchaseQuantity": 99,
"Name": "Green Pears Loose Class 1",
"OfferPromotion": "",
"OfferValidity": "",
"OfferLabelImagePath": "",
"Price": 0.37,
"PriceDescription": "£1.97 / kg",
"ProductId": "253624099",
"ProductType": "LooseWeightOrQuantityProduct",
"UnitPrice": 1.97,
"UnitType": "kg"
},
{
...next product....
}
]
}
```

Parameter Name	Parameter Value
COMMAND	LISTFAVOURITES
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

Parameter Name	Parameter Value
PAGE	(Optional) used to get a page of favourites rather than all of them (the customer may have hundreds!). See syntax details for ‘product pagination’.

LIST PENDING ORDERS

Purpose: Lists any pending orders - that is, orders which the customer has placed but which have not yet been delivered. You can use a forthcoming command to switch a customer's account to amend a pending order in a future update to the API.

Example:
http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=LISTPENDINGORDERS&sessionkey=ls9rsHv7FXdW1AseYnnWYQJIl4fzCe9w1Vd1MAqz2YsDmud3xA

Example returned JSON for a customer with one pending order:

```
{
"StatusCode": 0,
"StatusInfo": "",
"PendingOrders":
[
{
"DeliveryLocationDateTimeDescription": "Home on Thursday 20th May between
18:00 - 20:00",
"SlotDateTimeStart": "2010-05-20 18:00",
"SlotDateTimeEnd": "2010-05-20 20:00",
"GuidePrice": "£6.92",
"OrderNumber": "3220550",
"OrderReference": "13291294"
}
]
}
```

Parameter Name	Parameter Value
COMMAND	LISTPENDINGORDERS
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

LIST PRODUCT CATEGORIES

Purpose: Lists the departments, aisles and shelves that make up the category hierarchy. The ID from the lowest category level - the shelf - can be used with LISTPRODUCTSBYCATEGORY to list the products in that shelf.

Example:
http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=LISTPENDINGORDERS&sessionkey=ls9rsHv7FXdW1AseYnnWYQJIl4fzCe9w1Vd1MAqz2YsDmud3xA

Example returned JSON:

```
{
"StatusCode": 0,
"StatusInfo": "Command Processed OK",
"Departments":
[
{
    "Id": "1",
    "Name": "Fresh Food",
    "Aisles":
    [
        {
        "Id": "12",
        "Name": "Fresh Fruit",
        "Shelves":
        [
            {
            "Id": "13",
            "Name": "Apples"
            },
            {
            "Id": "14",
            "Name": "Pears"
            },
            {
            "Id": "15",
            "Name": "Citrus Fruit"
            }, ...etc
```

Parameter Name	Parameter Value
COMMAND	LISTPRODUCTCATEGORIES
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

LIST PRODUCT OFFERS

Purpose: Lists all products on offer. Page Pagination is available with this command.

Example:
http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=LISTPRODUCTOFFERS&page=1&sessionkey=ls9rsHv7FXdW1AseYnnWYQJIl4fzCe9w1Vd1MAqz2YsDmud3xA

Example returned JSON:

```
{
"StatusCode": 0,
"StatusInfo": "Command Processed OK",
"PageNumber": 1,
"TotalPageCount": 19,
"TotalProductCount": 1869,
"PageProductCount": 100,
"Products":
[
{
"BaseProductId": "59046587",
"EANBarcode": "5016325763600",
"CheaperAlternativeProductId": "",
"HealthierAlternativeProductId": "",
"ImagePath": "http://img.tesco.com/Groceries/pi/600/5016325763600/
IDShot_90x90.jpg",
"MaximumPurchaseQuantity": 99,
"Name": "Carlsberg 15X440ml",
"OfferPromotion": "Any 2 for £16.00",
"OfferValidity": "valid from 11/5/2010 until 16/5/2010",
"OfferLabelImagePath": "http://www.tesco.com/Groceries/UIAssets/I/Sites/
Retail/Superstore/Online/Product/pos/2for.png",
"Price": 12,
"PriceDescription": "£1.82 each",
"ProductId": "259636790",
"ProductType": "QuantityOnlyProduct",
"UnitPrice": 1.82,
"UnitType": "LITRE"
}, etc ...
```

Parameter Name	Parameter Value
COMMAND	LISTPRODUCTOFFERS
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.
PAGE	(Optional) used to return the requested page number.

LIST PRODUCTS BY CATEGORY

Purpose: Lists all products in a shelf category. Note that page pagination does not work this command - all products for the shelf are returned regardless of count.

Example:
http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=LISTPRODUCTSBYCATEGORY&category=18&sessionkey=ls9rsHv7FXdW1AseYnnWYQJIl4fzCe9w1Vd1MAqz2YsDmud3xA

Example returned JSON:

```
{
"StatusCode": 0,
"StatusInfo": "Command Processed OK",
"PageProductCount": 6,
"Products":
[
{
"BaseProductId": "64028383",
"EANBarcode": "0000010073971",
"CheaperAlternativeProductId": "",
"HealthierAlternativeProductId": "",
"ImagePath": "http://img.tesco.com/Groceries/pi/971/0000010073971/
IDShot_90x90.jpg",
"MaximumPurchaseQuantity": 99,
"Name": "New Season Strawberries", ...

etc ...
```

Parameter Name	Parameter Value
COMMAND	LISTPRODUCTSBYCATEGORY
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.
CATEGORY	A shelf category ID from LISTPRODUCTCATEGORIES command (department and aisle category IDs do not work).
EXTENDEDINFO	(Optional) use value “N” for basic info (equivalent to not supplying this parameter), or “Y” for extended information. **WARNING: Using ExtendedInfo=Y on searches with a large returned product count places a large burden on the API and will take a long time to respond. See ‘PRODUCT SEARCH - EXTENDED INFO BEST PRACTICE’.**

LOGIN
Purpose: Logs a customer into their Tesco grocery account and returns a session key which can be used with all the other commands so that the API knows which session a command applies to. Login also checks the developer and application keys to ensure that they are valid. Note that this command only works in secure (https) mode. The web browser and server send all parameters (after the domain (web site) name) securely in an encrypted data connection.
Example: https://secure.techfortesco.com/groceryapi_b1/restservice.aspx?command=LOGIN&email=nick@example.com&password=secret&developerkey=HgQuT6GHye3C6jklhj65&applicationkey=ECI987G56FF417663A05
Example returned JSON:

```
{
"StatusCode": 0,
"StatusInfo": "Command Processed OK",
"BranchNumber": "2431",
"CustomerId": "12592340",
"CustomerName": "Mr Lansley",
"SessionKey": "x38yJTParR282iuQrmvcmgBwLhwhLKJqKj6rcmxYy1WRR4j5me",
"ChosenDeliverySlotInfo": "No delivery slot is reserved.",
"CustomerMessageOfTheDay": ""
}
```

Parameter Name	Parameter Value
COMMAND	LOGIN
EMAIL	The customer’s login email address at the Tesco grocery service.
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.
PASSWORD	The customer’s password
DEVELOPERKEY	20-character alphanumeric string given to a developer at the developer’s portal at http://www.techfortesco.com/tescoapiweb
APPLICATIONKEY	20-character alphanumeric string also obtained at the developer’s portal.

PRODUCT SEARCH
Purpose: Executes searches for products on the grocery service, using text that can be part of a product's description, or 13-digit numeric barcode. Page pagination is available for this coma
Example: http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=PRODUCTSEARCH&searchtext=chocolate&page=1&sessionkey=JXM3UBD4FgpokEtbhmpCB9nT4TYCOcQz6LkfRKfVS33NRSuRE4

Example returned JSON:
`"StatusCode": 0,`
`"StatusInfo": "Command Processed OK",`
`"PageNumber": 1,`
`"TotalPageCount": 52,`
`"TotalProductCount": 1033,`
`"PageProductCount": 20,`
`"Products":`
`[`
`{`
`"BaseProductId": "59935480",`
`"EANBarcode": "8410510031524",`
`"CheaperAlternativeProductId": "",`
`"HealthierAlternativeProductId": "",`
`... etc ...`

Parameter Name	Parameter Value
COMMAND	PRODUCTSEARCH
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.
PAGE	(Optional) used to get a page of favourites rather than all of them (the customer may have hundreds!). See syntax details for 'product pagination'.
SEARCHTEXT	Text to search for products, 9-digit Product ID, or 13-digit numeric barcode value. If using characters that cannot be sent in an HTTP web address, you will need to "URLEncode" them. For example, spaces need to be converted either into "+" symbols or the %20 hex symbol. Most software development kits and computer languages support a URLEncode() method or function.
EXTENDEDINFO	(Optional) use value "N" for basic info (equivalent to not supplying this parameter), or "Y" for extended information. **WARNING: Using ExtendedInfo=Y on searches with a large returned product count places a large burden on the API and will take a long time to respond. See 'best practice' on the next page:**

PRODUCT SEARCH - EXTENDED INFO BEST PRACTICE
Purpose: Returns extended information about a product, using EXTENDEDINFO=Y
http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=PRODUCTSEARCH&SEARCHTEXT=258510116&extendedinfo=Y&sessionkey=CkccbhkjLEX92RG32Q8IAUFagDUDKnaXjJPKtkwMmgmXcMTkyl

It is easy for the API to supply all kinds of extended information about products, including Recommended Daily Allowance (RDA), ingredients list and nutritional information.

However it is important to request the API "wisely" for this data, because the amount of data returned can be large and the time required for the API to gather the information, format and return it to your client application can be lengthy.

Here is best practice for using this feature:

1) When your customer searches for products (whether by text search, barcode, listing favourites or offers), avoid using the ExtendedInfo parameter or set EXTENDEDINFO=N in the SEARCHTEXT command in order to get back basic information (which is quite considerable in itself, and includes links to images and any offer info where applicable).
2) Provide a facility for the customer to highlight a product, whereupon you supply the 9-digit product ID to the SEARCHTEXT parameter of the PRODUCTSEARCH command (see example above) and use EXTENDEDINFO=Y to return the extended information for that one product (see example JSON on next page).

On the next page you will see highlighted elements that you will find useful for your customers to see, although you must bear in mind that this data can vary wildly between products. On some products we don't include all extended info. For example on some sweet and chocolate products there is no RDA information because quite frankly you would be treating yourself and you don't want to spoil the moment by learning how deliciously bad the product is for you...!

What about searching for product that are "free-from" an ingredient or below a certain number of grammes of fat per portion?

Currently the API is unable to provide the filtering itself so in these circumstances it may well be necessary for you to request extended info for a product search. We accept that this is a useful feature and, going forward we aim the API to be able to do this itself. However only a production API with access to the product database will be able to achieve this. Until then one way of achieving best practice is to have your own server that uses our API and also processes requests form your client applications. You would design a service that would queue requests from the client application, perform the extended search, filter out those products not satisfying the nutrition or ingredients request, and then alert that application that the data was ready for it.

We have a continuous ongoing programme of improvements to product information so expect this data to improve over time as well as give the API fast filtering access to the product database which it does not have today for security reasons.

PRODUCT SEARCH - EXTENDED INFO BEST PRACTICE

Example returned JSON (truncated with "..." replacing extended textual info):

```
{
"StatusCode": 0,
"StatusInfo": "Command Processed OK",
"PageNumber": 0,
"TotalPageCount": 1,
"TotalProductCount": 1,
"PageProductCount": 1,
"Products":
[
{
"BaseProductId": "50020453",
"EANBarcode": "5018374350930",
"CheaperAlternativeProductId": "",
"CookingAndUsage": "...", <== Useful cooking and usage tips
"ExtendedDescription": "...", <== The manufacturer's blurb appears here
"HealthierAlternativeProductId": "258510145",
"ImagePath": "...",
"MaximumPurchaseQuantity": 99,
"Name": "Tesco Baked Bean In Tomato Sauce 420G",
"OfferPromotion": "3 for £1.00",
"OfferValidity": "valid from 1/3/2010 until 24/5/2010",
"OfferLabelImagePath": "...",
"Price": 0.44,
"PriceDescription": "£1.05 each",
"ProductId": "258510116",
"ProductType": "QuantityOnlyProduct",
"Rating": 0,
"StorageInfo": "...", <== Useful info on storage for this product
"UnitPrice": 1.05,
"UnitType": "kg",
"RDA_Calories_Count": "165",   <== The 5 RDA values on many products
"RDA_Calories_Percent": "8",   <== Both in count and percent
"RDA_Sugar_Grammes": "12",
"RDA_Sugar_Percent": "13",
"RDA_Fat_Grammes": "1",
"RDA_Fat_Percent": "2",
"RDA_Saturates_Grammes": "0",
"RDA_Saturates_Percent": "1",
"RDA_Salt_Grammes": "1.4",
"RDA_Salt_Percent": "23",
"NutrientsCount": 9,   <== count of nutrients
"Nutrients":
[
{
"NutrientName": "Energy",
"SampleDescription": "100g contain",
"SampleSize": "335kJ (80kcal) ",
"ServingDescription": "Half of a can (210g) contains",
"ServingSize": "700kJ (165kcal) "
} ... x 9 nutrients
],
"IngredientsCount": 10,   <== count of ingredients
"Ingredients":
[
{"Name": "Haricot Beans (49%)"},
{"Name": "Water"},
{"Name": "Tomato Pure (20%)"}, ... x 10 ingredients
]
}
]
```

READY FOR CHECKOUT
Purpose: Tests whether the current order is ready for checkout. There are two tests - one is that there are 5 or more products in the basket and that a delivery slot has been chosen. This is useful if you wish to inform the customer that they have done everything they need and checkout will be successful (checkout occurs outside the API).
Example: http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=READYFORCHECKOUT&sessionkey=JXM3UBD4FgpokEtbhmpCB9nT4TYCOcQz6LkfRKfVS33NRSuRE4
Example returned JSON for test success:

```
{
"StatusCode": 0,
"StatusInfo": "9 items in basket, and delivery slot reserved for delivery
between 2010-05-18 17:00 and 19:00 - Checkout OK to proceed.",
"ItemsInBasket": 9,
"DeliverySlotStart": "2010-05-18 17:00",
"DeliverySlotEnd": "2010-05-18 19:00",
"DeliverySlotReservationExpires": "2010-05-13 19:05",
"ReadyForCheckout": "Y"
}
```

Example returned JSON for test fail:

```
{
"StatusCode": 105,
"StatusInfo": "Fewer than 5 items in the basket",
"ItemsInBasket": 0,
"DeliverySlotStart": "0001-01-01 00:00",
"DeliverySlotEnd": "0001-01-01 00:00",
"DeliverySlotReservationExpires": "0001-01-01 00:00",
"ReadyForCheckout": "N"
}
```

Parameter Name	Parameter Value
COMMAND	READYFORCHECKOUT
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

SERVER DATE TIME
Purpose: Returns the API server's date and time both in local (UK) time and UTC / GMT. During British wintertime, both Local and UTC values are the same. Note: You do not have to have executed the LOGIN command before you use this command.
Example: http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=SERVERDATETIME

Example returned JSON:
`{`
`"StatusCode": 0,`
`"StatusInfo": "SUCCESS",`
`"ServerLocalDateTime": "10/27/2010 6:09:24 PM",`
`"ServerUTCDateTime": "10/27/2010 5:09:24 PM"`
`}`

Parameter Name	Parameter Value
COMMAND	SERVERDATETIME
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

SAVE AMEND ORDER
Purpose: Saves changes to the basket of an order being amended, and returns the session to ‘normal’ mode, restoring the contents of the basket that were there beforethe AMENDORDER command was used to switch the session into ‘Amend Order’ mode.
Example: http://www.techfortesco.com/groceryapi_b1/restservice.aspx?command=SAVEAMENDORDER&sessionkey=JYbMVtiaIPuGRWiHuPQxcSAGUKDpHoyw5l3U7L5fXF4c2Khlq2
Example returned JSON: `{ "StatusCode": 0, "StatusInfo": "Amend Order saved successfully" }`

Parameter Name	Parameter Value
COMMAND	SAVEAMENDORDER
SESSIONKEY	50-character session key
JSONP	(Optional) used to wrap the returned JSON data inside a function whose name is the value stated.

End of Documentation