PROGRAMMING BOTS FOR TIENDEO
MANUAL – Stores bots

version 2 (2013-11-19)

There are 2 kinds of bots for Tiendeo:

- "stores" bots: a stores bot parses a retailer's web page and find out all the stores belonging to the retailer in a given country (including information such as opening hours, location, etc.)
- "catalogs" bots: a catalogs bots parses a retailer's web page and find out all the catalogs available and valid for the retailer

This manual describes how to build a stores bot.

IMPORTANT: *Please notice that the bots must find data in the official(s) web site(s) of the retailer. It is forbidden to use other web sites.*

Table of contents

1/ Introduction

Let's take as an example the bots that were built for Carrefour Spain (a supermarket chain).
On Carrefour Spain web site (http://www.carrefour.es/) there is a page made to find a store:
http://carrefour.es/tiendas-carrefour/buscador-de-tiendas/
You first have to choose the region, and then appears a page containing the list of the stores located in the region. When you click on one of the store, you go to the store's web page where the robot will get all the information.

The aim of the bot is to parse these HTML pages and to store the information in an XML file in a standard format.

The ouput XML file generated by the bot is:

```
<?xml version="1.0" encoding="UTF-8"?>
<bot>
	<store>
		<retailer value="Carrefour" />
		<country code="ES" />
		<url value="http://carrefour.es/tiendas-carrefour/_tcmLinkFilter.aspx?tcmUri=3861" />
		<name value="Carrefour Almería" />
		<address value="Avda. del Mediterráneo, 244" />
		<postcode value="04006" />
		<town type="localidad" value="Almería" />
		<region type="provincia" value="Almería" />
		<phonenumber value="902202000" />
		<coordinates lat="36.84145717" lon="-2.443735735" />
		<description value="Festivos: de 10:00 a 22:00" />
		<id value="3861" />
		<openinghours weekday="1" from="09:00" till="22:00" />
		<openinghours weekday="2" from="09:00" till="22:00" />
		<openinghours weekday="3" from="09:00" till="22:00" />
		<openinghours weekday="4" from="09:00" till="22:00" />
		<openinghours weekday="5" from="09:00" till="22:00" />
		<openinghours weekday="6" from="09:00" till="22:00" />
	</store>
	<store>
		<retailer value="Carrefour" />
		<country code="ES" />
		<url value="http://carrefour.es/tiendas-carrefour/_tcmLinkFilter.aspx?tcmUri=3888" />
		<name value="Carrefour El Ejido" />
		<address value="Autovía A-7, km. 406" />
		<postcode value="04700" />
		<town type="localidad" value="El Ejido" />
		<region type="provincia" value="Almería" />
		<phonenumber value="902202000" />
		<coordinates lat="36.7680897" lon="-2.827794434" />
		<description value="Festivos: de 10:00 a 22:00" />
		<id value="3888" />
		<openinghours weekday="1" from="09:00" till="22:00" />
		<openinghours weekday="2" from="09:00" till="22:00" />
		<openinghours weekday="3" from="09:00" till="22:00" />
		<openinghours weekday="4" from="09:00" till="22:00" />
		<openinghours weekday="5" from="09:00" till="22:00" />
		<openinghours weekday="6" from="09:00" till="22:00" />
	</store>
	<!-- and many other stores... -->
</bot>
```

2/ Bots

A bot can be written in PHP (version 5.3), C#, or possibly in another language (in this case, please contact Tiendeo first). The source code must be provided.

As an output, every bot must create and write an XML file containing all the information it found. **It shall receive as an input the path of this XML file**.

If the bot takes more than 30 minutes to run completely in normal conditions, please make it possible to curve its running time by adding a *timeout* in **minutes**. The bot should act like this:

If the timeout is reached, the bot writes to the XML file all the stores it has found so far.

The bot will be launched by using a command line such as:

```
php.exe -f C:\mybot.php – C:\mybot_output.xml 60
```
(lauch php.exe so that it executes mybot.php. The bot will generates the file mybot_output.xml. Timeout: 60 minutes)

or

```
mybot.exe C:\mybot_output.xml 60
```
(lauch mybot.exe; it will generates the file mybot_output.xml. Timeout: 60 minutes)

The bot should be tested (see 4. Testing your bots). The execution time and the ouput of the tested bot should be provided as well.

In each country, you will have one folder, with all your bots and possible common classes. The name of the files (before extension) must not contain other characters than lower case letters (a, b, c, …) and the underscore character (_).

If you change a common class file or a bot file, please do not forget to send it as well.

Here are the things that should be provided for each bot you develop:

- the bot itself (including source code)
- the country, the name of the retailer or its webpage URL
- the command line necessary to launch the bot
- the time the bot ran in your test
- the XML output of your final test.

3/ XML output for stores bots

The bot output must be a valid XML 1.0 file. The first line of the file should be:
```
<?xml version="1.0" encoding="UTF-8"?>
```
The encoding is highly recommended to be UTF-8.

The root of the XML tree is called “bot” and its children are called “store”. Each “store” node contains the information the bot found about one store belonging to a retailer.
```
<?xml version="1.0" encoding="UTF-8"?>
<bot>
<store></store>
<store></store>
<store></store>
</bot>
```

3.1/ Necessary fields

Retailer (datatype: string)
The name of the retailer.
```
Example: <retailer value="IKEA"/>
```

URL (datatype: string)
The URL of the (closest) web page where the information about the store can be found. At best the URL of the store's web page if every store has its own page, at worst the web page of the retailer.
```
Example: <url value="http://www.ikea.com/gb/en/store/glasgow"/>
```

Country (datatype: string consisting of 2 letters)
The ISO 3166-2 code of the country where the store is located.
```
Example: <country code="GB"/>
```

Region (datatype: string)
The administrative zone (region, province, state, …) where the store is located. As this field highly depends on the country, more information should be given to you about this field.
```
Example: <region type="regiontype" value="regionname"/>
```

Town (datatype: string)
The town where the store is located. If the notion of town is abiguous in the considered country, please contact Tiendeo.
```
Example: <town value="Glasgow"/>
```

Address (datatype: string)
The address of the store (minus the country, region, town, and postal code which are saved in other fields). It usually contains the kind of route, the route name and the building's number in the street.
```
Example: <address value="99 Kings Inch Drive"/>
```

3.2/ Mandatory (if available) fields

Post code (datatype: string, format depending on the country)
The post code corresponding to the place where the store is located. The format of the string depends on the country.

```
Example: <postcode value="G51 4FB"/>
```

Coordinates (datatype: 2 decimal numbers)
The geographic coordinates (WGS84) of the store in degrees ("lat" for latitude, "lon" for longitude). Google Maps API shouldn't be used.

```
Example: <coordinates lat="55.870146" lon="-4.355031"/>
```

Phone number (datatype: string)
The phone number that can be called to get in touch with the store. One tag corresponds to one phone number but there could be several phonenumber tags (the most important appearing first the file). The phone number should be such as every one in the country could call and should thus include a possible region prefix.

```
Example: <phonenumber value="0845 355 2266"/>
```

ID (datatype: string)
A string ID for the store, such as one store keeps having the same ID even if the bot is launched several times.

```
Example: <id value="266"/> (IKEA UK website assigned an integer ID to all theirs stores)
```

Opening hours
The store is by default closed every day of the week. You can specify a time slot during which the store opens by adding an "openinghours" tag, containing:
- weekday: 1 for Monday, 2 for Tuesday, …, 7 for Sunday
- from: *hh:mm* format
- till: *hh:mm* format

The time should follow ISO 8601 (24-hour clock system) and should be given in the local time zone.

```
Example:    <openinghours weekday="5" from="09:00" till="18:00"/>
            <openinghours weekday="6" from="09:00" till="12:00"/>
            <openinghours weekday="6" from="14:00" till="20:00"/>
```

… means that the stores opens on Fridays from 9am till 6pm and on Saturdays from 9am till 12am and from 2pm till 6pm.

Shopping centre (datatype: string)
The name of the shopping center to which the store belongs.

```
Example: <shoppingcentre value="Braehead"/>
```

3.3/ Optional fields

Name (datatype: string)
The name of the store, if it has a specific name.

```
Example: <name value="IKEA Glasgow"/>
```

Description (datatype: *string*)
A possible description of the store, or other information that cannot be saved in other fields.

```
Example: <description value="Restaurant Opening Hours Breakfast: Monday - Friday 9.30am – 11am"/>
```

Parking (datatype: boolean *true* or *false*)
Whether the store owns a parking lot (car park) or not.

```
Example: <parking value="true"/>
```

E-mail (datatype: string)
An e-mail address that can be used to contact the store.
`Example: <email value="email@email.com"/>`

4/ Testing your bots

Every bot must be tested before it is sent to Tiendeo.
You must run the bot, and save the output file as well as write down the time necessary to its execution.

TO-CHECK LIST:

- the XML output file is correct
- every store seems to have a correct address
- one store (with one ID) does not appear more than once
- the necessary/mandatory fields are presents.
- the number of stores found seems correct
- you have randomly picked 10 stores on the web page and the information (every field) is correctly saved in the XML file for these stores

APPENDIX A: Country specific fields

SPAIN
Code (ISO 3166-2): ES
Postal code format: 5 digits \d{5}
Region “provincia”: <region type="provincia" value="..."/>
Region “communidad autonoma”: <region type="communidad" value="..."/>

ITALY
Code (ISO 3166-2): IT
Postal code format: 5 digits \d{5}
Region “provincia”: <region type="provincia" value="..."/>
Region “regione”: <region type="regione" value="..."/>

UNITED KINGDOM
Code (ISO 3166-2): GB
Postal code format: 2/3/4 characters + space + 3 characters
(see http://en.wikipedia.org/wiki/Postcodes_in_the_United_Kingdom#Validation)
Region “county”: <region type="county" value="..."/>
Region “country”: <region type="country" value="..."/>

MEXICO
Code (ISO 3166-2): MX
Postal code format: 5 digits \d{5}
Region “estado” (or DF): <region type="estado" value="..."/>

BRAZIL
Code (ISO 3166-2): BR
Postal code format: 5(+3) digits \d{5} *OR* \d{5}\-\d{3}
Region “UF” (federal unit): <region type="uf" value="..."/>

COLOMBIA
Code (ISO 3166-2): CO
Postal code format: 6 digits \d{6}
Region “departamento”: <region type="departamento" value="..."/>

ARGENTINA
Code (ISO 3166-2): AR
Postal code format: \w\d{4}\w{3}
(see http://en.wikipedia.org/wiki/Postal_codes_in_Argentina)
Region “provincia”: <region type="provincia" value="..."/>

INDIA
Code (ISO 3166-2): IN
Postal code format: 6 digits \d{6}
Region “state”: <region type="state" value="..."/>

FRANCE
Code (ISO 3166-2): FR

Postal code format: 5 characters \d(\d|\w)\d{3}
Region “département”: <region type="departement" value="..."/>
Region “région”: <region type="region" value="..."/>

UNITED STATES
Code (ISO 3166-2): FR
Postal code format: 5(+3) digits \d{5} *OR* \d{5}\-\d{3}
Region “state”: <region type="state" value="..."/>

NETHERLANDS
Code (ISO 3166-2): NL
Postal code format: \d{4}\s?\w{2}
(see http://en.wikipedia.org/wiki/Postal_codes_in_the_Netherlands)
Region “provincie”: <region type="provincie" value="..."/>