WEATHER CHECKING APPLICATION

Lech Dzięgielewski
Secondary School Degree, Zespół Szkół Elektrycznych im.T.Kościuszki w Opolu
E-mail: ldziegielewski@o2.pl

Supervised by: Michał Podpora
E-mail: ravyr@klub.chip.pl

ABSTRACT

In this paper author describes the development process of his computer application, capable of getting the weather forecasts data from definite www servers.

1. INTRODUCTION

Weather forecasting is one of these problems, which are never going to expire. People read forecasts in newspapers everyday, watch them in TV and check them in specialised Internet services. The possibility to check the weather in the Internet seems to be the most attractive one for software developers.

The main goal of my project was to create an application able to connect with one of these services, and to show the actual weather of any place on the world to the user.

2. DEVELOPING PROGRESS

The most important thing, when writing a computer program, is to decide, what exactly the program has to do, and how do you want to make it do it. In my case, the point was clear - I wanted to know what temperature is outside and how it is going to change in the next twelve hours.

I came to conclusion that the best solution will be to connect to one of the Internet services with cities database huge enough, and then to get the specified information.

2.1. FIRST STEPS

The programming language I used, was C#. This language is extremely intuitive and easy to understand. The .NET Framework platform gives developers the access to many useful classes and functions. In addition, .NET Framework manages the memory on its own, so developers don’t have to concern about e.g. forgotten, non-used variables.

The integrated development environment (IDE) I used, was Microsoft Visual C# 2008 Express Edition. Its biggest advantage is that it is free of charge as long as I don’t use it to create commercial programs. In addition, its complex but intuitive interface is great advantage for new developers.

Like I mentioned before - my program is connecting to the service over the Internet. To do it I used HttpWebRequest and HttpWebResponse classes - they are default .NET Framework classes. Thanks to them my program is able to connect with any website and obtain its HTML code. In this code, my program is going to search for information it needs.

An important advantage of this application (in comparison to weather forecast servers) is the time, which you need to get the information. In my application I focused only on certain, most important information, passing over lots of useless or redundant data. Using my application, user can get the information that he really needs faster than through any web browser. In addition, my application can be minimized to tray - the forecast for current city is auto-refreshed, so the user doesn't need to use web browser at all to get the information about weather.

2.2. VISUAL INTERFACE

What would be a good program without good interface? In my case, visual look is pretty simple. “Weather Checker” contains:

- one form - a place where the user can type the desired city name
- one submit button
- sixteen controls (“labels” and “pictureboxes”) used to show gathered information.

In case of more than one search result, a window containing “listbox” appears. User can choose which city from the list interests him.

Obrázek 1: Visual interface.

2.3. HTML CODE PROCESSING

“Weather Checker” is processing HTML code three times during one search. First and second time it searches through the code to find if any cities are found (I’ll explain later why it happens twice). If it finds a city, it adds it to the city list and opens the results window. The third time, it is processing the code to gather specific information:

- actual time
- actual temperature (it also shows “real feel” temperature – it’s conditioned by temperature, wind and humidity)
- weather (rain, snow, clouds etc.)

- humidity
- air pressure
- wind (direction and speed)
- visibility
- sunrise and sunset times
- next 12 hours forecast

To process the code, I used methods provided by String class – String.Contains, String.Replace, String.Remove, String.IndexOf and String.Insert. These methods are used to edit text strings – a special loop is reading HTML code line by line, and if any of the lines contain specific code, these methods are editing it and assigning values to the variables.

2.4. ENCOUNTERED PROBLEMS AND SOLUTIONS

I've already created few similar programs, but this project has been the largest I've ever made. This had to cause several new problems.

The biggest of all of them was very easy to solve. The service I used, offers two different sub pages - one is used for searching for USA cities, second for cities of "the rest of the world". The solution was the double use of HttpWebRequest and HttpWebResponse classes during the search operation.

In webpages that use CMS (Content Management Systems) or CSS (Cascade Style Sheets), the location and/or content of certain information in HTML code may be changed without changing overall site template (structure). My application is going to work correctly even after some serious changes of a site's content. Only full reorganization of service may cause some problems in Weather Checker searching process. Luckily, such things happen very rarely, because in case of huge services it takes a lot of time and work.

3. CONCLUSIONS

Final effect is satisfying. Program is showing current weather in any place of the world. During the development process, I broadened my knowledge and programming skills. In future I'm planning to add "next 7 days" forecast. I have also been thinking about converting "Weather Checker" into personal desktop organizer (clock, calendar with option to add notes, alarm, weather etc.). I hope I'll be able to realize my plans in future.

REFERENCES

[1] http://msdn2.microsoft.com/en-us/library

[2] http://www.codeproject.com

[3] Perry, S.: Core C# and .NET, Helion S.A., Warszawa 2006, ISBN 83-246-0320-4