# User Manual
# Microsoft Dynamics AX Add-on
# HTML Print

Version 1.2

Last update: 06.09.2010

# Contents

# 1 Introduction

Microsoft Dynamics AX reports are not very flexible when it comes to user definable content. The lack of design options that do not require application changes is a major weakness of AX reports.

The Dynamics AX HTML Print add-on provides the user with a tool to create content that can be printed on any report. Layout options include font styles, colours, backgrounds, frames, images and more.

All a programmer or power user has to do is adding a section control and two lines of code to the report. After that, any user may change the content with a built-in graphical editor or any external HTML editor.

# 2 Installation

The add-on is delivered as source code in an xpo file. The labels are contained in separate files. The code is based on the Dynamics AX version stated in the customer's purchase order.
The program shall be installed by an expert. If the application has already been adapted, the user might merge these with the add-on program code.

The xpo must be compiled on a machine where the HTML Print .NET component is installed (e.g. any AOS machine).

## 2.1 HTML Print .NET component

HTML Print requires a .NET component, which must be installed on every AOS machine by running *HTMLPrint.exe*. Earlier versions of the component must be uninstalled first. Before installing or uninstalling this component, all AOS must be stopped.

## 2.2 Problems

If incomprehensible problems arise after the installation of the program, it often helps to delete the client cache. The client cache files are located in the folder for local settings (usually C:\Documents and Settings\<user_account>\Local Settings\Application Data). The files are named "*.*auc*" in Dynamics AX 4.0/2009 and "*.*aoc*" in AX 3.0. They are given the "hidden" attribute. The files shall only be deleted if the client is not running.
If the problems persist, the application shall be compiled.

# 3 Setup

## 3.1 Configuration

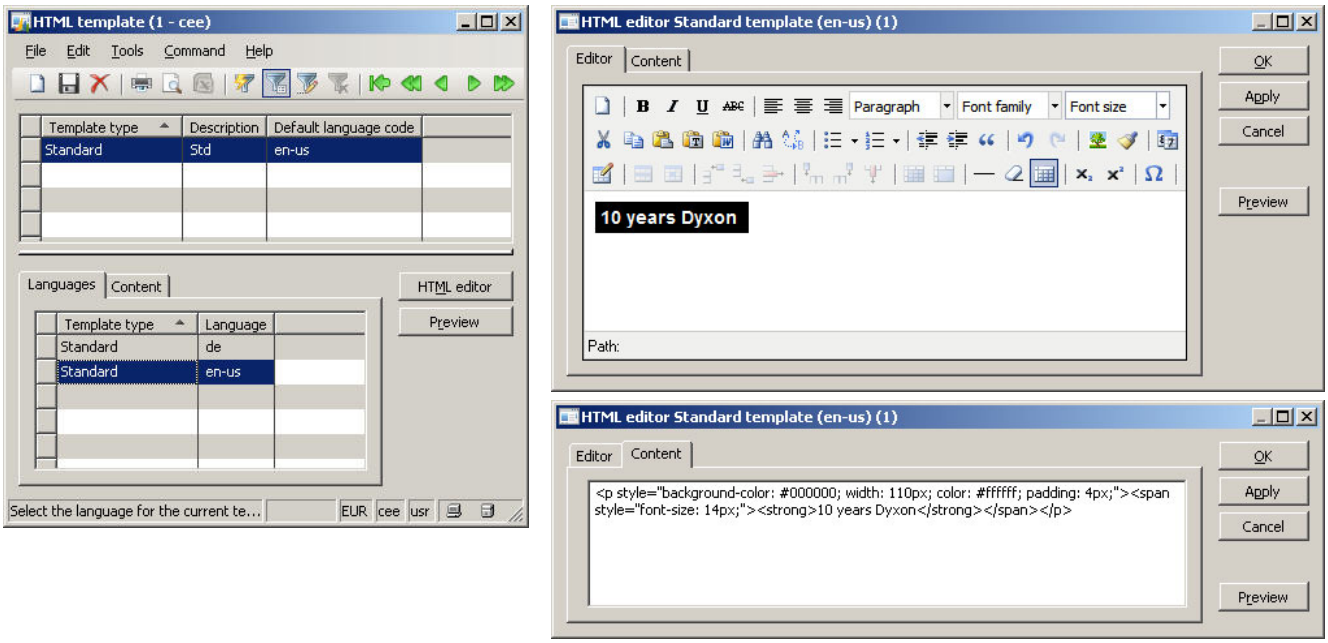
Activate configuration key *Dyxon Add-ons > HTML Editor* to enable the fully featured editor included with the add-on.

## 3.2 Security

Access to all add-on components can be configured under the *Basic* security key.

## 3.3 Templates

HTML content for this add-on can be created through templates, files or any database table field.
HTML print templates can be created from menu *Basic → Setup → HTML templates.*




Each template has a default language, usually the company's main language. Different HTML content may be created for different languages.
The content may be modified by opening the editor or by copying/pasting HTML code from an external editor.

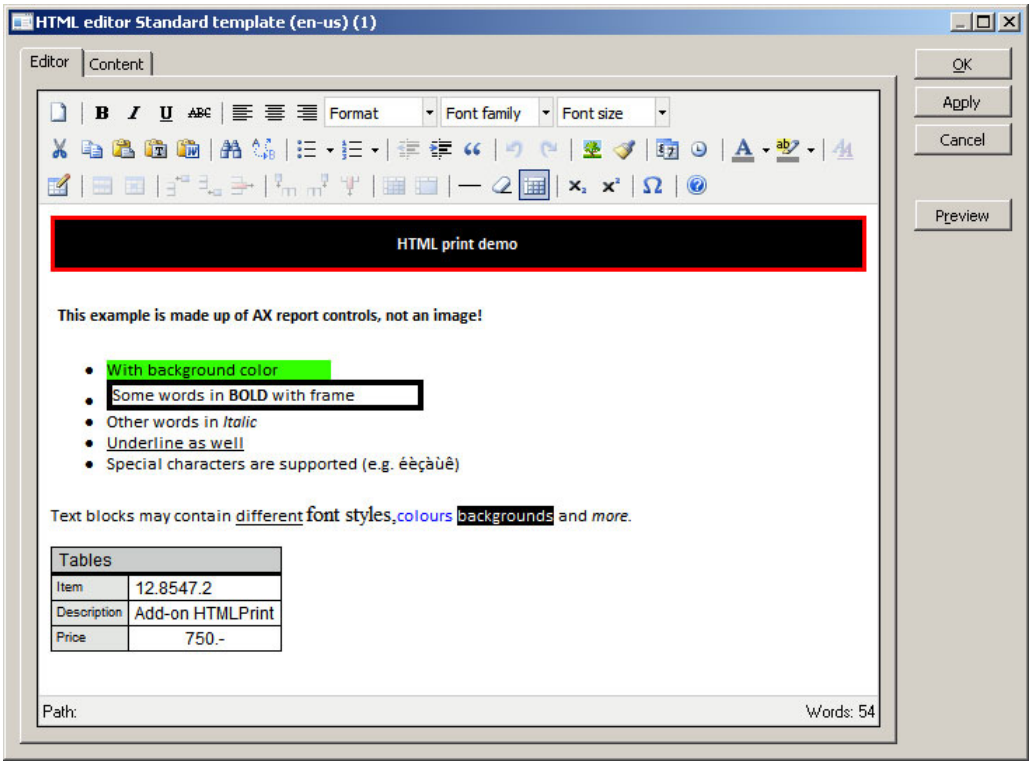
The preview button opens a report showing the content as report controls and as an image on two different pages. As on-screen previewing may show minor flaws, it is recommended to print the report onto paper to get accurate results.

## 3.4 Creating new template types

New template types (e.g. for purchase order document, sales invoice, etc.) may be created by extending the base enum *DYX_HTMLPrintTemplateType*.

# 4 HTML editor

A fully featured HTML editor is included with the add-on. Internet Explorer 7 or 8 is required and JavaScript has to be enabled for the editor.
Please disable configuration key *Dyxon Add-ons* > *HTML Editor* if the requirements are not met. The default Dynamics AX editor will then be used instead.

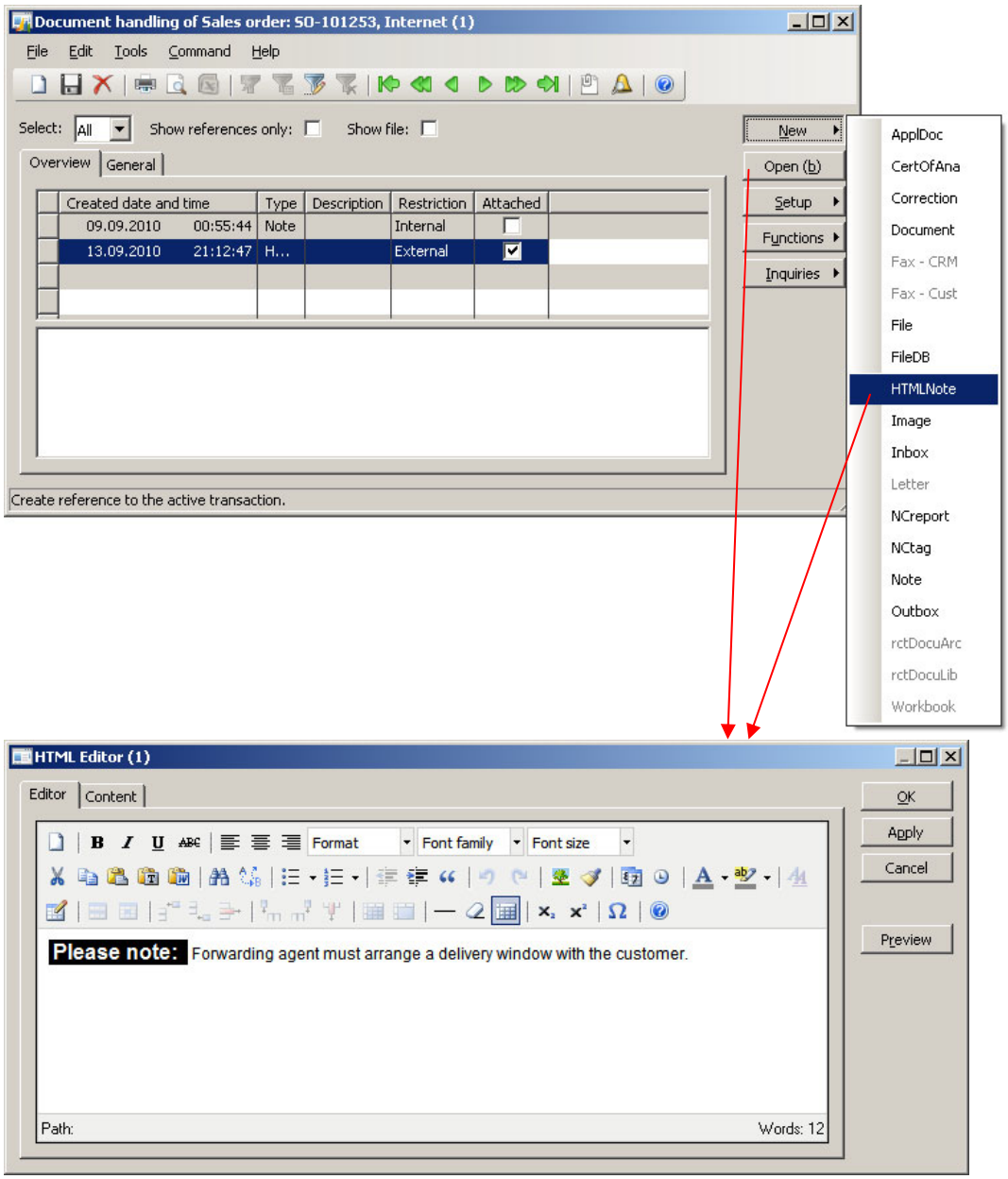



## 4.1 License / Copyright

The editor is based on TinyMCE, which is released under LGPL. See this page for details.

# 5 Document handling

A new document type has been created. This allows notes attached to any Dynamics AX record to be designed using the HTML editor. To setup the new type, open the document types form in menu *Basic* > *Setup* > *Document management*, create a new type record and select “HTML note” in the class description field.
Creating a new note or opening an existing one starts up the HTML editor.




See below for examples on how to enable reports to print HTML notes.

# 6 Adding content to reports

Adding HTML content to a report is straightforward and easy. Content can be printed as AX report controls or as an image. Using report controls is the preferred method, since print quality is better and text is still searchable (e.g. in a pdf file). Images may be required if the HTML output becomes too complex for report controls.

The *DYX_HTMLPrintExample* report, included in the add-on package, shows how to add HTML content.

## 6.1 “Report controls” method

Report controls are added by the HTML Print add-on. All that’s required is a report section control and HTML content:

1. Create a programmable section, or use an existing section.
2. In a suitable method, add code to create the content.
3. Execute the section.

Sample code, added to the *fetch* method:

```
DYX_HTMLPrint htmlPrint = DYX_HTMLPrint::construct(element);
htmlPrint.setSize100mm(15000, 12000);
htmlPrint.addElementsFromContent(ctrlSection,
DYX_HTMLPrintTemplate::findContent(DYX_HTMLPrintTemplateType::Default,
element.design().languageID()).HtmlContent);
element.execute(1);
```

### 6.1.1 “Report controls” method with page split

If a HTML section is longer than one page or if the unprinted bottom of a page shall be filled, HTML elements have to be spread over more than one page.
Below is a code sample that fills the page and moves the remaining elements to the next page:

```
DYX_HTMLPrint htmlPrint = DYX_HTMLPrint::construct(element);
int heightLeft;

htmlPrint.setSize100mm(20000, 0);
htmlPrint.initElementsFromContent(
DYX_HTMLPrintTemplate::findContent(DYX_HTMLPrintTemplateType::Default,
element.design().languageID()).HtmlContent);
heightLeft = element.mm100Left() - ctrlSection.topMarginAndFrame() -
ctrlSection.bottomMarginAndFrame() - element.heightOfPageFooters();
while (htmlPrint.hasMoreElements())
{
    heightLeft = element.mm100Left() - ctrlSection.topMarginAndFrame() -
    ctrlSection.bottomMarginAndFrame() - element.heightOfPageFooters();
    htmlPrint.addNextElements(ctrlSection, heightLeft);
    element.execute(1);
}
```

## 6.2 “Image” method

1. Create a programmable section, or use an existing section.
2. Add a display method which returns a report variable of data type bitmap.
3. Add a bitmap to the section, where the *DataMethod* property is the method name created above.
4. In a suitable method, add code to create the content.
5. Execute the section.

Sample code, added to the *fetch* method:

```
DYX_HTMLPrint htmlPrint = DYX_HTMLPrint::construct(element);
htmlPrint.setSize100mm(15000, 12000);
htmlPrint.setResolution(600);
// bitmapData is variable of type bitmap, declared in class declaration
bitmapData = htmlPrint.createImageFromContent(
DYX_HTMLPrintTemplate::findContent(DYX_HTMLPrintTemplateType::Default,
element.design().languageID()).HtmlContent);
// ctrlImg is the image control
ctrlImg.resizeBitmap(false);
ctrlImg.top100mm(htmlPrint.getImageOffsetY100mm());
ctrlImg.left100mm(htmlPrint.getImageOffsetX100mm());
ctrlImg.width100mm(htmlPrint.getImageWidth100mm());
ctrlImg.height100mm(htmlPrint.getImageHeight100mm());
element.execute(2);
```

## 6.3 Document handling

Follow these steps to print HTML notes stored in document handling:

1. Create a programmable section, or use an existing section.
2. The following code will print all external notes attached to record “refRecord”:

```
DYX_DocuActionHTMLNote::printNotes(element,
                                   programmableSection,
                                   DocuRefSearch::newTypeIdAndRestriction(
                                         refRecord,
                                         “”,
                                         DocuRestriction::External));
```

This code will print all external notes attached to a packing slip onto a section with name “HTMLNotes”:

```
DYX_DocuActionHTMLNote::printNotes(element,
                                   HTMLNotes,
                                   DocuRefSearch::newTypeIdAndRestriction(custPackingSlipJour,
                                   custFormletterDocument.DocuTypePackingSlip,
                                   DocuRestriction::External));
```

## 6.4 Restrictions

All files must be accessible by the AOS service account from the AOS machine. This includes html content files and files referenced from html content (e.g. images or style sheets).
The parser of this add-on is not fully HTML and CSS compliant. However, most HTML tags and CSS formatting options are supported. Reports should always be previewed before using HTML code in live environments.

## 6.5 API description

### 6.5.1 Class DYX_HtmlPrint

**static DYX_HtmlPrint construct(ReportRun _reportRun)**
Creates an object of the class *DYX_HtmlPrint*

**void setSize100mm(int _width, int _height)**
Size of printable area in hundredth of mm. Default value for width: Printable width for selected printer. Default value for height: Printable height left on current page.
Size should always be specified to prevent from unpredictable result.

**void setResolution(int _resolution)**
Resolution in dpi (only required for method “image”)

**void setCache(boolean _cache)**
Use caching (default: true). Disable cache if html content changes for each printout. Cached data is stored in table *DYX_HTMLPrintCache*.

**void setOffset100mm(int _x, int _y)**
Offset (_x = left, _y = top) for html elements on section in hundredth of mm (default: 0, only for method “report controls”).

**void setOptimize(boolean _optimize)**
Optimize size (default: true, only for method “image”). Output image gets clipped to reduce memory consumption.

**void setThrowIfError(boolean _throwIfError)**
Abort if an error occurs in the html parser, show error message if false (default: true)

**void addElementsFromContent(ReportSection _section, str _htmlContent)**
Add report controls to section, created from html content.

**void addElementsFromFile(ReportSection _section, str _fileName)**
Add report controls to section, created from file content.

**void initElementsFromContent(str _htmlContent)**
Prepare report controls, created from html content.

**void initElementsFromFile(str _fileName)**
Prepare report controls, created from file content.

**boolean hasMoreElements()**
Check for more elements to be added (use together with addNextElements)

**void addNextElements(ReportSection _section, [int _maxHeight100mm, boolean _autoNewPage])**
Add report controls to section, prepared in one of the “initElementsFrom…” methods. Parameter maxHeight100mm (optional) specifies the printable height left (in hundredth of mm) on the current page. If parameter autoNewPage (optional) is set to true (default value), a page break is inserted if no element fits onto the current page.

**Bitmap createImageFromContent(str _htmlContent)**
Create bitmap image from html content.

**Bitmap createImageFromFile(str _fileName)**
Create bitmap image from file content.

**int getImageHeight100mm()**
Get height for created image in hundredth of mm. Only different from height set in method setSize100mm if optimization is enabled.

**int getImageWidth100mm()**
Get width for created image in hundredth of mm. Only different from width set in method setSize100mm if optimization is enabled.

**int getImageOffsetX100mm()**
Left offset for created image in hundredth of mm. Only higher than zero if optimization is enabled.

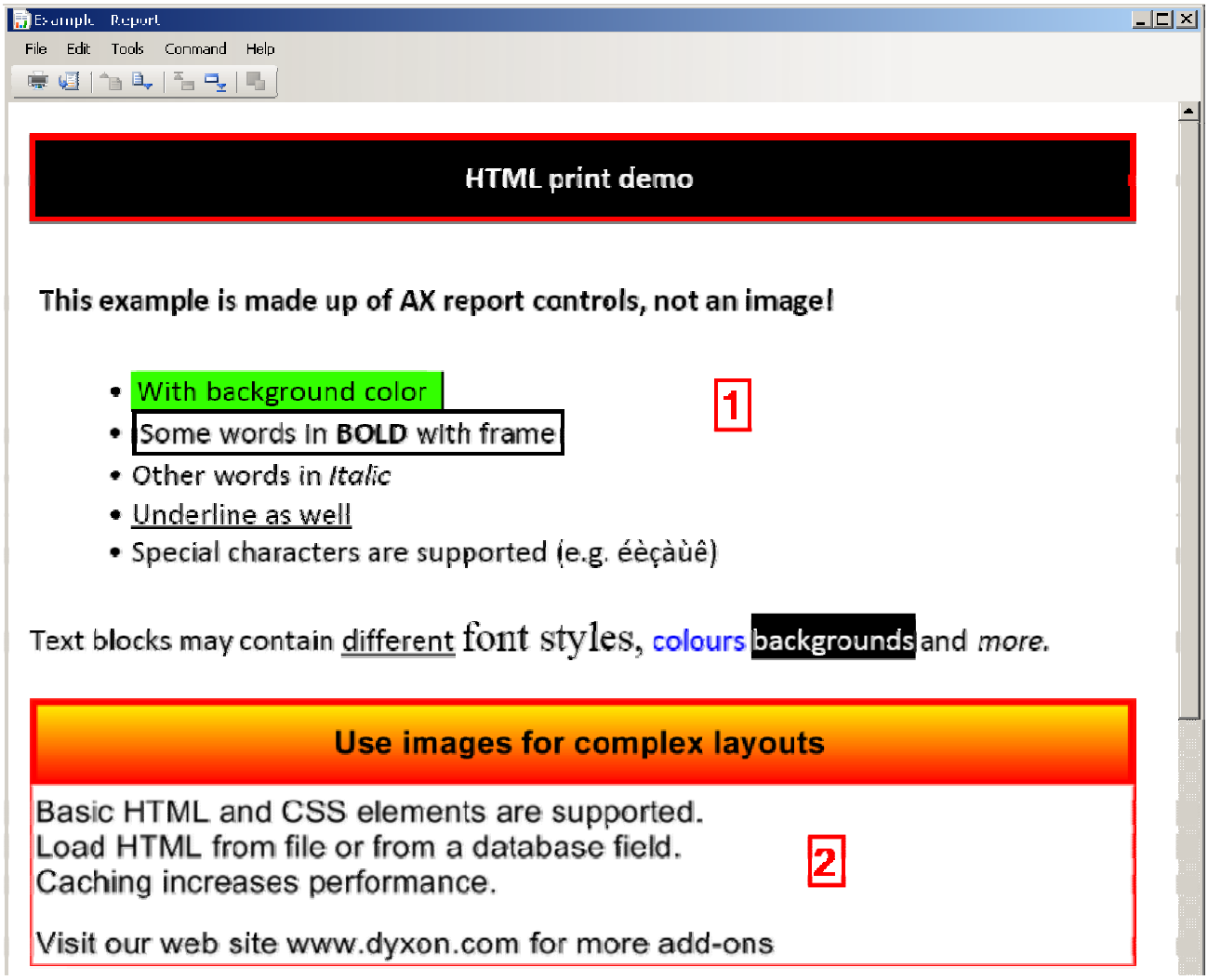
**int getImageOffsetY100mm()**
Top offset for created image in hundredth of mm. Only higher than zero if optimization is enabled.

### 6.5.2 Class DYX_DocuActionHTMLNote

**static str getHTMLContentFromDocuRef(DocuRef _docuRef)**
Get HTML content as string from DocuRef

**static void printNotes(ReportRun _reportRun, ReportSection _section, DocuRefSearch _search)**
Print notes (DocuRef) returned by DocuRefSearch onto the ReportSection.

# 7 Samples




## 1: Printed as AX controls

HTML print demo
This example is made up of AX report controls, not an image! <ul>
<li> With background color </li>
<li> Some words in  BOLD with frame </li>
<li> Other words in Italic </li><li> <u>Underline as well</u> </li>
<li> Special characters are supported (e.g. éèçàùê) </li></ul>
Text blocks may contain <u>different</u>
font styles,colours
backgrounds and more.

## 2: Printed as image, text is converted to image

<div style=”border: thin red solid; font-family:Arial, Helvetica, sans-serif”><div style=”background-color: yellow; background-gradient: red; text-align:center;padding:10px; border:3px red solid”>Use images for complex layouts Basic HTML and CSS elements are supported.<br>Load HTML from file or from a database field.<br>Caching increases performance.<br><br>Visit our web site www.dyxon.com for more add-ons