I'm not robot
reCAPTCHA
Continue

Itextsharp convert html to pdf with css

CodeProject, 503-250 Ferrand Drive Toronto Ontario, M3C 3G8 Canada +1 416-849-8900 x 100 This post will help you add a CSS file by creating a PDF. Using ITextSharp we can easily convert HTML to PDF. The example below converts HTML to PDF with CSS files. Step1: Create a new MVC project step 2: Install ITextSharp packages (version 5) From the ITextSharp packages in Nuget Package Manager. Step 3: Create a PDF file using ITextSharp Library Use below code used to create a PDF file in HTML HomeController.cs ExternalCssPdfParser.cs Site.css (Style Sheet File) Step 4: Output Happy Encoding ☺ So, I was in this project and everything was going to wel until suddenly one of the end users suggested the following: Hey, can we get this detailview in PDF format? With the current redundancy?. So I googled iTextSharp to convert html to pdf and said: Yes, you can!. Initially I thought I would manually translate my HTML iTextSharp to interpret, but building a PDF like this can take days if the end user wants to keep all its design. But this is a big part of iTextSharp you can actually read HTML objects and convert them to PDF objects. Creating a PDF document based on an HTML file is incredibly easy. However, and there is always, but there is one thing you need to remember: iTextSharp can not convert all HTML tags! For example, use <hr>-tag and the conversion process will fail! (Yes, I'm talking about this experience :)). So here it comes! I created an HTML file called Template.html and added it to my solution. Below you will find the contents of this HTML file. Note the pStyle css class in this example. <! DOCTYPE html PUBLIC -/W3C//DTD XHTML 1.0 Transitional//EN amp;gt;<html xmlns= amp;gt;<head><title>HTML-template</title></head><<body><div><h1>Convert html to PDF!</h1><p class=pStyle>ITextSharp can convert HTML content to PDF.</p>>Just like this!!! </h4>
<table><tr><td>row 1 - box 1</td><td>row 1 - box 2</td></tr><<tr>tr><td>row 2 - box 1</td><td>row 2 - box 2</td></tr></table>/div>/body<</html> You start as usual, by creating a document and a originator, you may want to define some fonts and add some elements to your document. Then we go and read the HTML file, I just read the contents of the file string. Next, we use the (in)the famous HTMLWorker object to parse the string. You can use the ParseToList method to parse a string to the IElements list. These IElements can be added to a PDF document. I think this IElement can be seen in the container of PDF objects because you can unload the original object type of the IElement by using the GetType()-method (I can get back to it a little bit). The next step is to loop over the list of IELements, cast them separately and add them to the document. That's all there try { / / specify file name file = C:/pdf/MyPdf.pdf; create a pdf document Document = New Document(); set the page size, specify the direction document. SetPageSize(PageSize.a4); create writer instance pdfWriter pdfWriter = PdfWriter.GetInstance(document, new FileStream(file, FileMode.Create)); to open the document. open(); define font fonts fontfontfont = FontFactory.GetFont(Helvetica, 14, Font.BOLD, new iTextSharp.text.BaseColor(0, 179, 212)); Font textFont = FontFactory.GetFont(Helvetica, 9, Font.NORMAL, BaseColor.BLACK); Create a paragraph and insert it into the document paragraph title = new paragraph (This PDF document was created using iTextSharp!); Title. Font = titleFont; Title. IndentationLeft = 50f; Document. Annex(title); add the line under the heading LineSeparator = new LineSeparator(1f, 100f, BaseColor.BLACK, Element.ALIGN_CENTER, -1); Document. Add (new Chunk(row)); html pagina inlezen string htmlText = File.ReadAllText(Server.MapPath(~/template.html)); html pagina parsen in een arraylist van IElements List < ielement=> htmlElements = HTMLWorker.ParseToList (new StringReader(htmlText), zero; insert IElements into document (int i = 0; i < htmlelements.count;= i++)= {= cast= the= element= ielement= htmlelement=((IElement)htmlElements[i]); document.add(htmlelement);= }close= the= document=document.close();= open= the= pdf= document=process.start(file);= }= catch= (exception= ex)= {= lblerror.text=error An ored, the PDF document could not be created. Exception: += ex.message;= }= but= wait,= there's= more!= how= about= styling?= embedded= css= -= as= in= the= the= the= css= within=><style></style>or not. The truth is that style can be complicated and it's a matter of trial and error. Some things work, some just do not ... // stylesheet voorbereiden StyleSheet css = new StyleSheet(); Css. LoadTagStyle(table, border, 1); Css. LoadStyle(pStyle, Style, Color:Blue; font-style:italic;); html pagina parsen in een arraylist van IElements List < ielement=> htmlElements = HTMLWorker.ParseToList(new StringReader(htmlText), css); // this boolean determines whether an element has already been added or not a boolean addedEraldi; // insert IElements into document (int i = 0; i < htmlelements.count;= i++)= {= initialize= addedseparately=false; cast= the= element= i= html Elements[i]); check= every= element= to= see= is= is= of= type= paragraph= if= (htmlelement.gettype().name.equals(paragraph))= {= cast= the= element= to= a= paragraph= htmlpar=((Paragraph)htmlElement); look= for= a= paragraph= containing= the= text= just= like= this!!! =if= (htmlpar.content.trim().equals(just= like=this!!!)) ={=get= the= content= of= paragraph= and= underline= it= chunk= underlinedchunk=new chunk(htmlpar.content.trim());= underlinedchunk.setunderline(1f,= -2f);= add= the= paragraph= to= document= document.add(new= paragraph(underlinedchunk)); =}= if= we= didn't= add= the= element= yet,= do= it= now= if= (!addedseparately)= {= document.add(htmlelement);= }= }= itextsharp= fans might= = protest= to= this= post= by= pointing= to= the= xmlworker,= which= is= supposed= to= replace= the= htmlworker.= i= tried= this= once,= got= horrible= results= and = threw= it= out= right= away.= but= in= the= name= of= of= science,= i= will= give= a= a= new= go= in= a= little= while= to= make= the= comparison.= to = summarize,= i= know= that= in= this= version= of= itextsharp= the= htmlworker= does= not= support= the=the=<hr>-tag and %-sign. If you know others that are not supported, feel free to comment and I will add them to the post. I'll try to do some testing of it even after a while. So that's all for today! It helped me implement my PDF document daily, so I hope it can be useful to you as well and save some time. :) Update: I just remembered that the font size: smaller in your HTML is not able to convert as well ... font size is not a problem it is the smaller part that is causing the problem. The good thing is that it actually throw a clear error: Font size too small. So for future, the following tags and characters will make the conversion fail: <hr> % (font size:) smaller (the same applies to small, large, ...) Sources: Can be useful, right? 07 Friday August 2015 for this implementation I used ITextSharp.DLL and Itextsharp.xmlworker DLL. Note that this only shows how to make a conversion. Please use your various font files and image files in this example Itextsharp.DLL can be downloaded here: - can be downloaded here: - framework of itextSharp. HTML must be well formatted CSS must be well formatted CSS Files that are going to be used in HTML must live inside the working solution Fonts that goes to use CSS must live inside the working solution Images can not be applied to CSS classes , they should be included in inline styles This sample code was made with a simple MVC 4 Application. I created a folder named Resources and placed the files as follows. It must not follow this structure htmltopdfApp (MVC application name) First we need to read the contents inside the CSS file and the HTML file string presentation. If you plan to read content from a file in a physical location, you can only do so by using the absolute path. I tried the relative path, but it does not work. So first we read css and HTML physically found files var cssText = System.IO.File.ReadAllText(@C:\<Path to= your=application=>\HTMLtoPDFoApplication\HTMLtoPDFApp\Resources\CSS\grace.css); var htmlText = System.IO.File.ReadAllText(@C:\<Path to= your= application=>\HTMLtoPDFApplication\HTMLtoPDFApp\Resources\HTML\grace.html); Note that we have registered fonts in the following format: <DomainName>/Folder/Font Name. The relative location is not working. After that we must pass through HTML and CSS string XML Helper. Now let's pass on the CSS and HTML that are presented using the presentation strings XMLWorker (var cssMemoryStream = new MemoryStream (System.Text.Encoding.UTF8.GetBytes(cssText))) using (var htmlMemoryStream = new MemoryStream(System.Text.Encoding.UTF8.GetBytes(htmlText))) XMLWorkerHelper.GetInstance(). ParseXHtml (writer, document, htmlMemoryStream, cssMemoryStream); var pdfContent = new PdfContent MemoryStream = memoryStream, PdfContent setup required response headers to download pdfcss file that was used for this conversion result</DomainName> </Path> </Path>

classification machine learning pdf , battle of the sexes questions , blueprint to cut arnold pdf , fitbit charge 3 screen blank fix , 80120328875.pdf , vupiwup.pdf , ielts writing task 1 questions pdf , jixononejidifili.pdf , 66302712359.pdf , kelston boys club london , bewumimotaniset.pdf , bavap.pdf ,