Stripping HTML

Wednesday, November 29, 2006

Ever wondered how to get just the text out of an html document, e.g. for indexing? NSAttributedString simply does not do it as you cannot stop it from loading external resources like images - resulting in timeouts and wasted network bandwidth.

Here is how we do it in our email client ('Ginko'). After a long research and asking around at WWDC, I was delighted that libxml2 can do everything needed and is very flexible, so the code is very short! Use it in your code as you like!

If someone improves it so that 'alt' tags etc. are also indexed, we'd like to update our code as well. // // NSString+OPHTMLTools.m // GinkoVoyager // // Created by Dirk Theisen on 30.06.06. // Copyright 2006 Objectpark Group. // #import "NSString+OPHTMLTools.h" #include #include #include @implementation NSString (OPHTMLTools) static void charactersParsed(void* context, const xmlChar* ch, int len) /*" Callback function for stringByStrippingHTML. "*/ { NSMutableString* result = context; NSString* parsedString; parsedString = [[NSString alloc] initWithBytesNoCopy: (xmlChar*) ch length: len encoding: NSUTF8StringEncoding freeWhenDone: NO]; [result appendString: parsedString]; [parsedString release]; } - (NSString*) stringByStrippingHTML /*" Interpretes the receiver als HTML, removes all tags and returns the plain text. "*/ { int mem_base = xmlMemBlocks(); NSMutableString* result = [NSMutableString string]; xmlSAXHandler handler; bzero(&handler, sizeof(xmlSAXHandler)); handler.characters = &charactersParsed; htmlSAXParseDoc((xmlChar*)[self UTF8String], "utf-8", &handler, result); if (mem_base != xmlMemBlocks()) { printf("Leak of %d blocks found in htmlSAXParseDoc", xmlMemBlocks() - mem_base); } return result; }

Have Fun!

Dirk