Task-Aware Location-Based Services for Mobile Environments

FP7-SME-207-1-222292-TALOS

Content Adaptation Technology D3.1

Deliverable lead contractor: CTI

Alexandros Efentakis	efedakis@cti.gr
Anna Stathaki	stathaki@cti.gr
Dieter Pfoser, CTI	pfoser@cti.gr

Due data: 30.8.2009
Actual submission date: 13.11.2009

Abstract
Presents (i) design of a content repository, (ii) a methodology for converting digital content to the repository, (iii) a Web interface for accessing content and geocoding tools.

Table of Contents

1 INTRODUCTION 5

2 CONVERTING CONTENT 5

2.1 DOCX FILE OVERVIEW 6

2.2 DOCX FILES PREPARATION AND SIMPLIFICATION 7

2.3 PARSING THE DOCX FILES AND INTERMEDIATE STORAGE 10

2.4 DATA MODEL AND IMPLEMENTATION 13

3 WEB INTERFACE 18

3.1 INTRODUCTION 18

3.2 SYSTEM ARCHITECTURE 18

3.3 DESCRIPTION OF THE WEB APPLICATION 25

3.4 PUBLIC PART 25

3.5 LIMITED ACCESS PART 26

3.6 CONCLUSIONS 34

1 Introduction

An essential aspect in TALOS is flexible and efficient content management. To that respect, this deliverables outlines (i) how to store existing content in a flexible manner and (ii) to provide a simple means for its manipulation and presentation.

One of the prerequisites of the TALOS project is creating a content repository that will be fully linked with the task hierarchy. Having content readily available from a project partner most of the approach focuses travel guides provided by TALOS partner MMV. However, any content source could be used in the process.

In addition, to provide for a flexible means of content manipulation, a Web interface was developed to presents the content stored in the database, allows for editing, and more importantly its geocoding. The interface provides for authentication and respective user rights management.

The following sections first present the content manipulation techniques, i.e., how to arrive from digital content sources at structured information stored in a database and subsequently describes the Web interface.

2 Converting Content

Being the only publishing partner, MMV provided us with the type of content they want to store in a universal repository. However, while the following methodology is described in terms of the specific content, it is universally applicable for any (digital) content source.

Each MMV travel guide is comprised from many docx files (native Microsoft Word 2007 format). Therefore, it was crucial to have a solid knowledge of the relatively new docx format.

MMV mainly deals with printed travel guides (although it features some electronic guides as well). Therefore its travel guides are mainly focused for printing purposes. From its beginning MMV used Microsoft Word for its DTP jobs. Consequently, the travel guides provided for the TALOS project are in Microsoft Word 2007 format (docx files). The docx format is the first attempt of Microsoft, towards a purely XML based format for storing office documents. This format is relatively new, so very few developers have actually developed applications for docx creation and manipulation. Also, since it is created by a commercial company like Microsoft, it is faced with mixed emotions from the open source community which favours the ODF format (the Open Office XML based format for storing office documents). One of the challenges of the TALOS project is to deal with this rather new format.

As a result, the TALOS workflow must use these docx files in order to fill the TALOS content repository. This process is roughly illustrated in the diagram below:

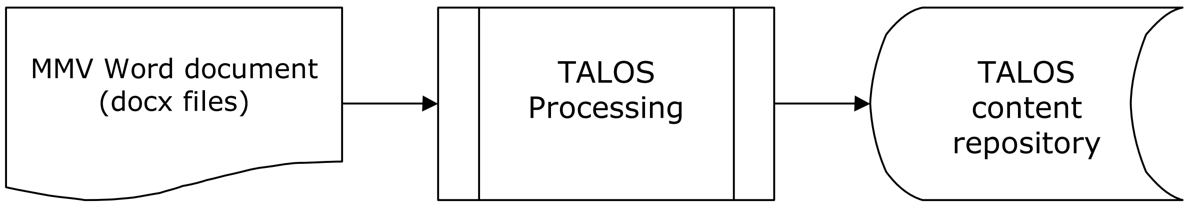


Figure 1: TALOS workflow rough description

Of course this rough description of the TALOS workflow includes several intermediate steps with varied levels of complexity. Those intermediate steps, along with the content repository data model will be described in the following sections.

2.1 Docx file overview

Docx files are based on the Office Open XML standard (also referred to as OOXML, or Open XML) created by Microsoft. OOXML is not only limited to Microsoft Word files but is also a file format for representing Excel spreadsheets or PowerPoint presentations. A docx file (like all Office Open XML files) is a ZIP-compatible package containing XML documents along with binary files, such as image files, along with a specification of the relationships between them.

A basic docx file contains an XML file called [Content_Types].xml at the root, along with three directories: _rels, docProps and word). The word directory contains the document.xml file which is the core content of the document.

The [Content_Types].xml file provides MIME type information for parts of the package, using defaults for certain file extensions and overrides for parts specificied by IRI.

The _rels directory contains relationships for the files within the package. To find the relationships for a specific file, look for the _rels directory that is a sibling of the file, and then for a file that has the original file name with a .rels appended to it. For example, if the content types file had any relationships, there would be a file called [Content_Types].xml.rels inside the _rels directory.

The _rels/.rel file is where the package relationships are located. The Microsoft Office applications look here first. When viewed in a text editor, one will see it outlines each relationship for that section. In a minimal document containing only the basic document.xml file, the relationships detailed are metadata and document.xml.

The docProps/core.xml file contains the core properties for any Office Open XML document.

As stated previously the word/document.xml file is the core content of any Word document.

So, basically in order to store a docx file in the TALOS repository, a custom Java application had to be built that reads specific XML files stored inside the docx file.

2.2 Docx files preparation and simplification

A word document (represented by a docx file) has a wealth of formatting information. Most of this formatting information is significant such as paragraph and character styles, because styles are used (or at least should be used) for identifying different types of content, such as chapter, section and subsection titles, image captions, TOC, indexes. In other cases Word uses formatting for identifying certain elements (such as hyperlinks, email links, prices, addresses) or strictly for visual purposes (bold, italics). This kind of formatting information is crucial and therefore should be also stored on our content repository.

On the other hand, some of the Word formatting information is of no actual structural use, such as spell checking marks, blank lines, tables, headers and footers, column breaks, section breaks, page breaks, comments, reviewing remarks and so-on. Unfortunately Microsoft Word also stores all this Word specific formatting information in the 'document.xml' file inside the docx file and therefore there is no easy way to separate useful from non-useful formatting commands.

Consequently, the process of parsing the docx files and inserting their content on the content repository should preserve all necessary structural information and remove all unnecessary Word specific formatting, which is bloated and has no actual use outside the Word application. Since we do not want to alter the original documents, we will create a 'simplified' version of each docx file that will be stored on a separate file. Once all 'simplified' versions of the original docx files are created, these are the files we will parse by our custom Java parser and not the original docx files, which will remain untouched.

This process can be described in the following diagram:

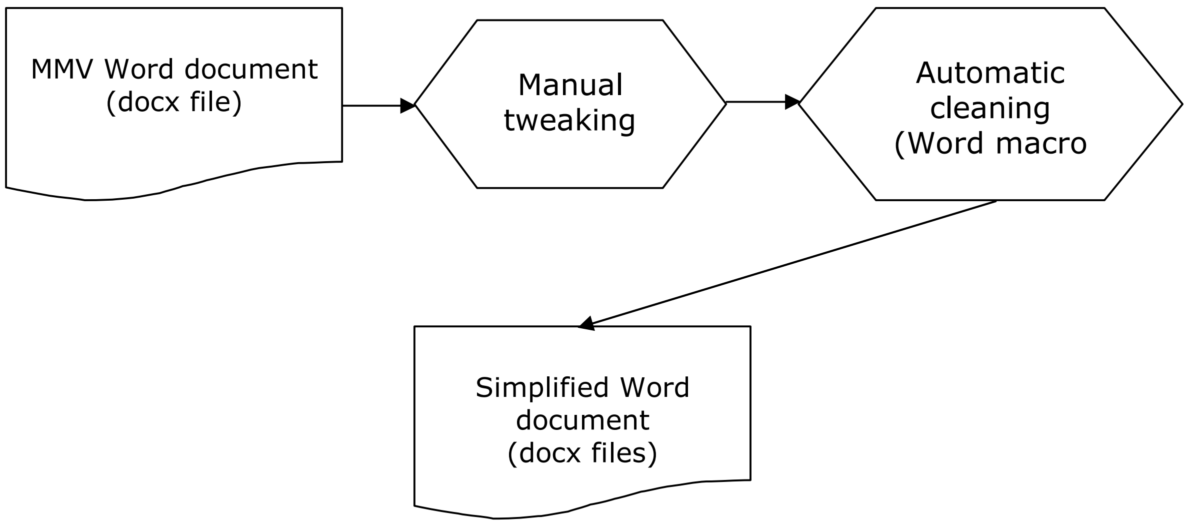

MMV Word document (docx file)
Manual tweaking
Automatic cleaning (Word macro
Simplified Word document (docx files)

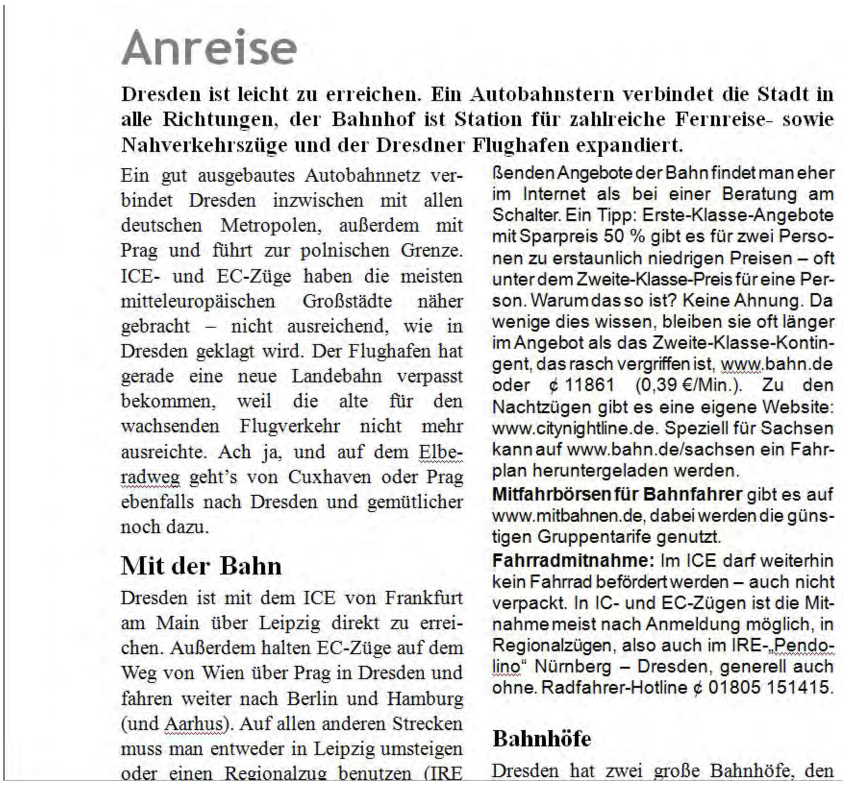
Figure 2: Preparation of docx files

The 'manual tweaking' phase is required to correct inconsistencies in the use of styles by the document authors. This is a common problem in all DTP companies. Although publishers always impose specific rules (which require the use of specific styles throughout the whole publishing organization) for formatting all kinds of business documents, these rules are rarely followed entirely by the authors or graphic designers. This is also a major problem with MMV documents, because each author is also responsible for the formatting of the travel guide he writes and therefore a central external interference (by the publisher) that could flatten inconsistencies in the use of styles is rather minimal.

After the manual tweaking phase and in order to automate the process of removing all unnecessary formatting information from the docx files, a Word macro was developed. What does this macro do?

- Opens all docx files on a given directory
- Removes comments, reviewing marks, spelling information, headers and footers, blank lines, hyphenation, section, column, page breaks
- Converts multicolumn content to just one column
- Convert tables to text
- Removes all direct character and paragraph formatting (formatting that is not done through proper use of styles)
- Stores the result as a separate doc file, with a certain suffix.

The result is much cleaner docx files which are easier to parse (as illustrated on the following picture).



Anreise

Dresden ist leicht zu erreichen. Ein Autobahnstern verbindet die Stadt in alle Richtungen, der Bahnhof ist Station für zahlreiche Fernreise- sowie Nahverkehrszüge und der Dresdner Flughafen expandiert.

Ein gut ausgebautes Autobahnnetz verbindet Dresden inzwischen mit allen deutschen Metropolen, außerdem mit Prag und führt zur polnischen Grenze. ICE- und EC-Züge haben die meisten mitteleuropäischen Großstädte näher gebracht – nicht ausreichend, wie in Dresden geklagt wird. Der Flughafen hat gerade eine neue Landebahn verpasst bekommen, weil die alte für den wachsenden Flugverkehr nicht mehr ausreichte. Ach ja, und auf dem Elberadweg geht's von Cuxhaven oder Prag ebenfalls nach Dresden und gemütlicher noch dazu.

Mit der Bahn

Dresden ist mit dem ICE von Frankfurt am Main über Leipzig direkt zu erreichen. Außerdem halten EC-Züge auf dem Weg von Wien über Prag in Dresden und fahren weiter nach Berlin und Hamburg (und Aarhus). Auf allen anderen Strecken muss man entweder in Leipzig umsteigen oder einen Regionalzug benutzen (IRE

ßenden Angebote der Bahn findet man eher im Internet als bei einer Beratung am Schalter. Ein Tipp: Erste-Klasse-Angebote mit Sparpreis 50 % gibt es für zwei Personen zu erstaunlich niedrigen Preisen – oft unter dem Zweite-Klasse-Preis für eine Person. Warum das so ist? Keine Ahnung. Da wenige dies wissen, bleiben sie oft länger im Angebot als das Zweite-Klasse-Kontingent, das rasch vergriffen ist, www.bahn.de oder ¢ 11861 (0,39 €/Min.). Zu den Nachtzügen gibt es eine eigene Website: www.citynightline.de. Speziell für Sachsen kann auf www.bahn.de/sachsen ein Fahrplan heruntergeladen werden.

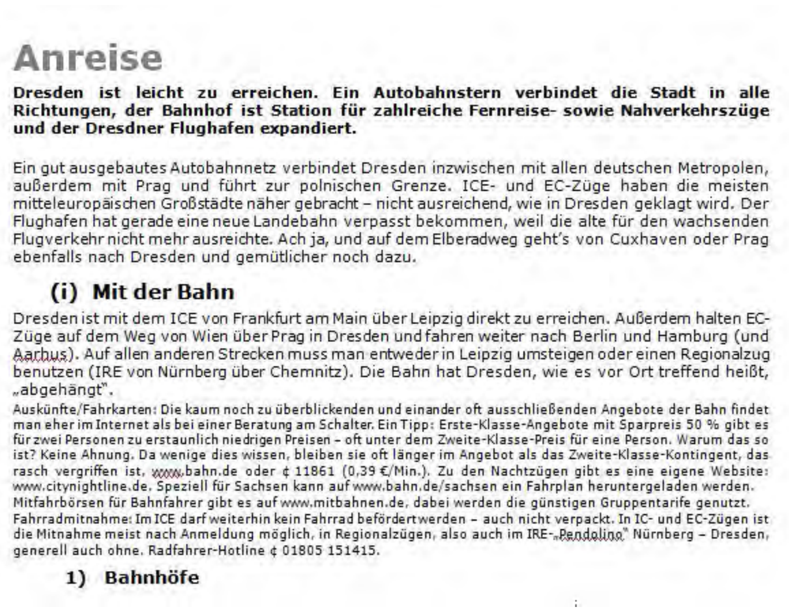
Mitfahrbörsen für Bahnfahrer gibt es auf www.mitbahnen.de, dabei werden die günstigen Gruppentarife genutzt.

Fahrradmitnahme: Im ICE darf weiterhin kein Fahrrad befördert werden – auch nicht verpackt. In IC- und EC-Zügen ist die Mitnahme meist nach Anmeldung möglich, in Regionalzügen, also auch im IRE-„Pendolino" Nürnberg – Dresden, generell auch ohne. Radfahrer-Hotline ¢ 01805 151415.

Bahnhöfe

Dresden hat zwei große Bahnhöfe, den

Figure 3: The word document in its initial form



Anreise

Dresden ist leicht zu erreichen. Ein Autobahnstern verbindet die Stadt in alle Richtungen, der Bahnhof ist Station für zahlreiche Fernreise- sowie Nahverkehrszüge und der Dresdner Flughafen expandiert.

Ein gut ausgebautes Autobahnnetz verbindet Dresden inzwischen mit allen deutschen Metropolen, außerdem mit Prag und führt zur polnischen Grenze. ICE- und EC-Züge haben die meisten mitteleuropäischen Großstädte näher gebracht – nicht ausreichend, wie in Dresden geklagt wird. Der Flughafen hat gerade eine neue Landebahn verpasst bekommen, weil die alte für den wachsenden Flugverkehr nicht mehr ausreichte. Ach ja, und auf dem Elberadweg geht's von Cuxhaven oder Prag ebenfalls nach Dresden und gemütlicher noch dazu.

(i) Mit der Bahn

Dresden ist mit dem ICE von Frankfurt am Main über Leipzig direkt zu erreichen. Außerdem halten EC-Züge auf dem Weg von Wien über Prag in Dresden und fahren weiter nach Berlin und Hamburg (und Aarhus). Auf allen anderen Strecken muss man entweder in Leipzig umsteigen oder einen Regionalzug benutzen (IRE von Nürnberg über Chemnitz). Die Bahn hat Dresden, wie es vor Ort treffend heißt, „abgehängt".

Auskünfte/Fahrkarten: Die kaum noch zu überblickenden und einander oft ausschließenden Angebote der Bahn findet man eher im Internet als bei einer Beratung am Schalter. Ein Tipp: Erste-Klasse-Angebote mit Sparpreis 50 % gibt es für zwei Personen zu erstaunlich niedrigen Preisen – oft unter dem Zweite-Klasse-Preis für eine Person. Warum das so ist? Keine Ahnung. Da wenige dies wissen, bleiben sie oft länger im Angebot als das Zweite-Klasse-Kontingent, das rasch vergriffen ist, www.bahn.de oder ¢ 11861 (0,39 €/Min.). Zu den Nachtzügen gibt es eine eigene Website: www.citynightline.de. Speziell für Sachsen kann auf www.bahn.de/sachsen ein Fahrplan heruntergeladen werden.
Mitfahrbörsen für Bahnfahrer gibt es auf www.mitbahnen.de, dabei werden die günstigen Gruppentarife genutzt.
Fahrradmitnahme: Im ICE darf weiterhin kein Fahrrad befördert werden – auch nicht verpackt. In IC- und EC-Zügen ist die Mitnahme meist nach Anmeldung möglich, in Regionalzügen, also auch im IRE-„Pendolino" Nürnberg – Dresden, generell auch ohne. Radfahrer-Hotline ¢ 01805 151415.

1) Bahnhöfe

Figure 4: The simplified word document after execution of the Word macro

The aforementioned process revealed several issues:

- The document provider (MMV) has not always used separate styles, for identifying different kinds of structural elements (eg. The style Heading 2 is used both for chapter and section headings).
- Sometimes, styles are not used at all. Instead the document author just applied direct character formatting (bold, colors, font size). This kind of information is lost and cannot be evaluated, since there is no way to tell if a bold portion of text is an email, or a hotel name, since they share the same formatting.

Therefore it is crucial for the document author to:

- Use a limited set of predefined styles for all his documents
- Use different styles for different types of content
- Avoid the use of direct paragraph and character formatting if possible, and ONLY ON TOP of existing styles.

For the sample travel guides provided by the MMV partner, things were not that discouraging as far as the first two cases are concerned (styles were pretty consistent on most cases), so a little manual tweaking was pretty sufficient to do the job.

After the simplification of the docx files, the docx files are ready for manipulation. This process will be described in the subsequent section

2.3 Parsing the docx files and intermediate storage

As stated previously, a docx file is basically a ZIP package containing various XML files. The word/document.xml is the the core content of any Word document. A typical travel guide also contains images, which are binary files. The information about which binary files are used in the Word document is stored inside word/_rels/document.xml.rels. So, basically parsing a docx document means parsing only those 2 xml files (word/document.xml and word/_rels/document.xml.rels) inside the docx package. That simplifies our work significantly.

Another issue that must be pointed out is that Office Open XML does not use mixed content but uses elements to put a series of text runs (element name r) into paragraphs (element name p). The result is terse and highly nested in contrast to HTML, for example, which is fairly flat, designed for humans to write in text editors and is more congenial for humans to read. On the other hand, XML files that do not contain mixed content are easier to parse and therefore the standard Java parser SAX2 is adequate for performing this task, without the need for any external Java libraries.

A travel guide consists of many docx files. It is assumed that for each travel guide, we create a separate folder for putting all the docx files associated with the specific travel guide.

Converting XML files to RDBMS has always been a very lively subject in the IT community. Instead of directly mapping the Word XML files to our data model, a new hybrid solution was suggested. Our approach is based on the fact that an XML document is basically a tree like structure of elements. Therefore storing the complete structure of an XML document (regardless of the actual XML tags) requires only 2 RDBMS tables. One for elements and one for attributes.

PostgreSQL the RDBMS of choice for TALOS (more on why PostgreSQL was selected will be mentioned on the data model section), features a Ltree module which may be used for describing tree like structures. Consequently we have all the necessary tools to describe the tree like structure of the XML document in our RDBMS.

This process is described on the following diagram:

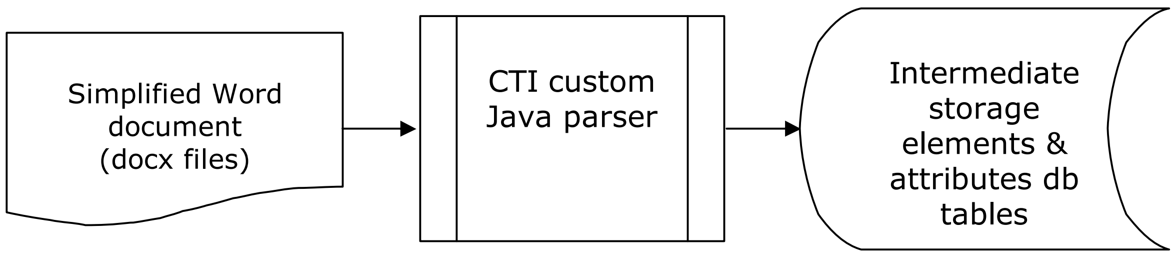

Simplified Word document (docx files)
CTI custom Java parser
Intermediate storage elements & attributes db tables

Figure 5: Parsing the docx files

2.3.1 The elements table

The elements table has the following fields:

- id
- element_path
- element_level
- element_name
- element_value
- word_path text
- word_file text
- xml_path text
- xml_file text

Elements are assigned a unique id (field id) per document (basically a counter that increments when a new XML element-tag opens). This is the **id** field that fully identifies an element within an XML document.

Each element record has also an **element_path** field that describes the full element 'nestingness'. For example an element with id=156 may have an element path field like '1.45.89.156' meaning that the element 156 is inside element 89, element 89 is inside the element 45 and element is inside element 1 (element with id 1 is always the root element of the XML document and therefore all XML elements path have 1 as their first 'ingredient').

The **element_level** field describes how deeply nested is the XML element. In our previous example the element 156 has an element_level of 4.

The **element_name** is the name of the XML element-tag (e.g w:p for paragraphs)

The **element_value** is the value of the XML element. For example, OOXML stores text inside w:t tags (eg. <w.t>This is my text</w.t>. So 'w:t' is the element name and 'This is my text' is the element_value).

The **word_path** is the folder where the docx file is located (can be used to identify different travel guides)

The **word_file** is the name of the docx file.

The **xml_path** is the folder where the XML file is located inside the docx package (For example the word/document.xml has an xml_path value of 'word'.

The **word_file** is the name of the XML file, within the docx file (eg. document.xml).

As mentioned before, PostgreSQL features an Ltree module that is appropriate for querying tree-like structures. The field element_path is of Ltree datatype and can be gist-indexed. Therefore querying the XML documentin order to get all paragraphs' text (paragraph is the w:p tag and text is stored in w:t tag) can be broken in 2 simple subqueries. Find the ids of all elements with element_name='w:p' and find all elements with element_name='w:t' that are 'children' of the paragraphs element (the Ltree module can execute such queries).

2.3.2 The attributes table

The attributes table has the following fields:

- element_id
- seq_indx
- attribute_name
- attribute_value
- word_path
- word_file
- xml_path
- xml_file

The **element_id** links the attribute with the element in the 'elements' table that this attribute belongs.

The **seq_indx** holds the information if this attribute is the first or second attribute for a specific element. Although, altering the attributes sequence within an element is permitted in most XML schemas, it is better to keep this information for covering even worst case scenarios.

The **attribute_name** is the name of the XML attribute

The **attribute_value** is the value of the XML attribute. For example <w:pStyle w:val="Heading1"/>. So 'w:val' is the attribute name and 'Heading1' is the attribute_value).

2.3.3 Storing OOXML in the tree-like RDBMS format assessment

Our Ltree approach is not RDBMS neutral and does not follow standard SQL specifications. However, other RDMBS have similar strategies for describing tree-like structures (Oracle has tree extensions simulating similar functionality), so therefore our approach could also be ported to some other RDBMS, if needed. On the other hand, we could directly query the XML documents through XQuery, but that would require 2 different databases. One pure XML database for storing the XML documents and one relational database that holds our data model. Although, some RDBMS (like Oracle or SQL Server) feature an XML datatype, that can be queried through XQuery as well, storing a whole XML document as only one record, would be slower and we still had to combine 2 query languages (SQL and XQuery instead of just SQL).

Storing XML documents as tree-like structures in a RDBMS, has also the obvious advantage that if the raw data is provided in some other XML format, there is nothing that needs to be changed in our approach. We will use the exact same Java parser and we simply have to slightly alter the SQL queries and stored procedures to extract the kind of information we need.

Another rather obvious advantage of storing the whole XML structure of the document in our RDBMS, is that we can extract the information we want at any abstraction level. For example, in our proposed data model (which will be described in the next section), paragraph is our lowest level structural unit. If we later choose, that we should choose a smaller structural unit (such as a sentence or even a phrase), there will be no need to go back to the original docx documents, we simply have to rearrange our SQL queries to provide us with the chosen level of detail.

As a conclusion, by using Ltree and SQL queries and taking advantage of the Ltree capabilities of PostgreSQL, we could extract any information from the XML document directly from our intermediate storage db tables. By using stored procedures, we can automate those SQL queries in order to automatically fill our TALOS content repository from our intermediate storage db tables, elements and attributes.

This process can be seen in the following diagram

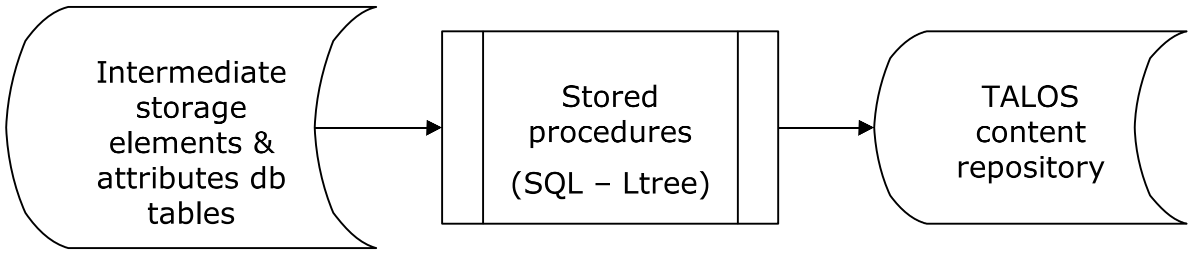


Figure 6: Filling the TALOS content repository from the elements and attributes table

In the following section we will discuss about our TALOS data model.

2.4 Data model and implementation

2.4.1 Introduction

Although travel guides have a lot of different thematic sections, they are written in way to be enjoyable and easy to read, than be treated as raw pieces of data. Thus, as even the TALOS partner MMV suggested, it is very difficult to create a thematic data model that will cover all travel guides. Travel guides are written by creative authors, each following his unique style of writing and preferences. The publishers do not want to impose limitations on the authors' creative writing, because that would affect the quality and readability of the final result and usually the publisher's clients were attracted to the publisher's book, due to the quality of the author's work. Therefore no publisher wants to 'fix something that is not broken' and impose limitations on the author's creativity.

That means that we have very diverse travel guides (even when written by the same author) because they are written either a) in distant periods of time or b) by different authors. Even when travel guides are re-published (every 2 years) they are not re-written from scratch but they are simply revised in order to keep up with changes in hotels, public transport, new museums or other places of interest.

Therefore trying to create a thematic model that will cover all travel guides will prove to be (if even possible) a very time consuming task. Therefore, we focused our effort to create a purely structural data model with paragraph as the lowest level structural unit. A book is a collection of paragraphs after all, in which some paragraphs have special structural meaning, such as chapter and section titles. We must also keep in mind that the travel guides were provided in Microsoft Word format, which is a DTP format describing a book's layout and not in some hierarchical XML or RDBMS format. Therefore we could not automatically extract thematic information that simply does not exist from a simple DTP layout.

There is also another issue. The only way to identify special paragraphs (inside the Word file) that have a special meaning, such as chapter or section titles, was by the use of special paragraph styles (like in HTML <h1> identifies 'chapters', <h2> identifies 'sub-chapters' or 'sections', <h3> identifies 'sub-sections' and so-on). Therefore for each paragraph we should also store their 'Word' style, because that is what identifies those special paragraphs

Unfortunately 3 new issues came up

- Two travel guides do not necessarily share the same set of styles. Therefore chapter titles on one travel guide are represented from a different paragraph style than chapter titles on the second travel guide.
- The same travel guide does not always make consistent use of the existing styles. For example a 'heading 3' style may be used to describe an actual 'heading 5'section title. That happens because a) the authors are not disciplined enough b) Microsoft Word is not a Content Management System, to enforce certain limitations on the authors.
- In very few cases, styles are not used at all. For example chapter titles are not described by some 'Heading 1' style but with direct paragraph formatting (font size 18, bold).

The first problem means that we have to study each travel guide and extract the set of styles used.

The third problem requires some manual tweaking directly on the Microsoft Word file (this step is done on the 'manual tweaking phase' of the preparation of MMV docx files).

The second issue has to be resolved inside our data model. Our data model should keep track of the current nesting level. For example Chapter titles are level 1, sub-chapters or sections are level 2 and subsections are level 3 and so-on.

In order to link paragraphs to the section (defined by their section title) we should keep for each paragraph its parent section information.

A paragraph may also include (besides text) an image (only one image per paragraph was encountered in the travel guides we were provided with). Therefore we should also keep the type of content (text or image) stored in a paragraph, along with the image information (the actual image file).

Initially there was also no need to store the sequence of paragraphs within a section with a separate field, because the Word file (=docx file = word/document.xml file) is in fact a sequential text file and therefore paragraphs are assigned (by our Java parser) unique auto-increment ids when they are stored. Thus, the paragraph ids actually include the paragraph sequence within the docx file. Perhaps (on a later phase and for strictly implementation reasons) we could add a 'sequence' field which will be automatically generated from the paragraph ids.

All those results can be summarized in the following section:

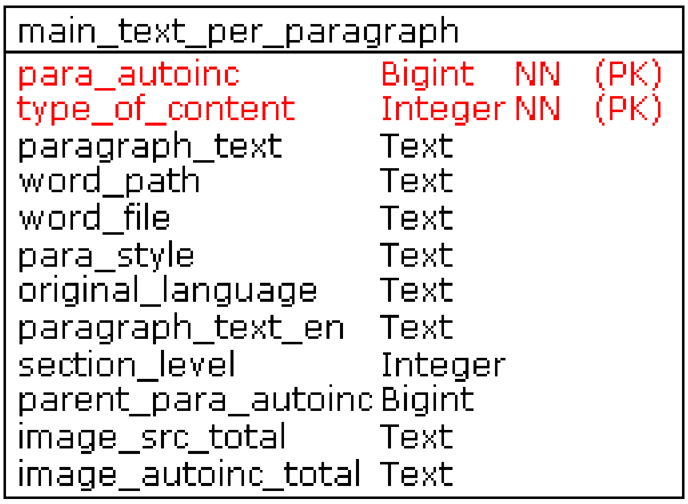
2.4.2 Textual description of data model

- Paragraph is the lowest structural unit we will use.
- Each paragraph is assigned a unique id
- Each paragraph may have text and an image. Therefore we need to store the type of content inside a paragraph (is it text or image?), the actual text and the image file information.
- Each paragraph has a paragraph style that may be used for identifying chapter titles, section titles and so-on).
- For each paragraph that holds chapter or section title we should have the level information describing how deeply nested this section title is, within the book hierarchy.
- Each paragraph belongs to the section title paragraph encountered immediately before the specific paragraph. Therefore for each paragraph we should store the parent section title that the specific paragraph belongs. If the paragraph represents a section title, its parent is the chapter title that this section belongs and so-on. This recursive model will be able to cover all cases, regardless of the book hierarchy.
- We should store the folder that the docx file existed (as we said before all docx files that belong to a specific travel guide are stored inside a separate folder and therefore the folder information links the paragraph to the actual travel guide).
- We should store the name of the docx file (so it will be easy to cross check the content of the paragraph with the Word file from which the paragraph was extracted).

2.4.3 Data model

From what is described in the previous section, only one RDBMS table is sufficient to store the structural hierarchy of a book. Although this may sound an over-simplified abstraction, the simplicity of our data model will be very easy to handle during the repurposing of the Travel guides

This is the ER diagram of our data model



main_text_per_paragraph			
para_autoinc	Bigint	NN	(PK)
type_of_content	Integer	NN	(PK)
paragraph_text	Text		
word_path	Text		
word_file	Text		
para_style	Text		
original_language	Text		
paragraph_text_en	Text		
section_level	Integer		
parent_para_autoinc	Bigint		
image_src_total	Text		
image_autoinc_total	Text		

Figure 7: The ER of our paragraph data model

The **para_autoinc** field is the unique id assigned per paragraph

The **type_of content** field may take the values '1' if we are talking about the text content of the paragraph, or '2' if we are talking about the image content of the paragraph.

The **paragraph_text** field holds the actual text of the paragraph. This field will have a value of null for all records with type_of content=2.

The **word_path** field holds the folder where the docx file resided. From that field we could separate the content of 2 travel guides, since the docx files of one travel guide are stored in a different folder than the docx files of the second travel guide.

The **word_file** field holds the name of the original docx file

The **para_style** field holds the paragraph style used on the docx file for each paragraph. This field may be used for identifying chapter titles, section titles and so-on.

The **paragraph_text_en** field holds the actual text of the paragraph translated in English. This translation is provided by a translation module, developed by CTI for a previous project (CITER – http://citer.cti.gr). The CTI translation module uses Google AJAX Language API (http://code.google.com/intl/el-GR/apis/ajaxlanguage/) in order to provide translation for various languages. Since, all travel guides were provided in German it is crucial to have some kind of English translation (even if it is not perfect), in order to understand the meaning of the paragraph.

The **original_language** field holds the original language (=German) of the docx file. This field was necessary for the CTI translation module

The **section_level** field holds the nesting level of section titles. Therefore this field holds a value of null for simple main text paragraphs. For chapter titles value = '1', for sub-chapters=sections value ='2', for subsections value='3' and so-on.

The **parent_para_autoinc** field describes in which section this specific paragraph belongs. This field can only take values already existing in the **para_autoinc** field, since it links recursively paragraphs with their parent paragraph.

The **image_src_total** field is the file reference to an image file used on the Word file. This field holds a value of null for all records with type_of content=1.

The **image_autoinc_total** field is an autoincrement counter for the images encountered. It is just a way to uniquely identify images.

2.4.4 Implementation

We decided to use PostgreSQL for our RDBMS uses. PostgreSQL is a powerful, open source object-relational database system. It has more than 15 years of active development and runs on all major operating systems, including Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), and Windows.

It is fully ACID compliant, has full support for foreign keys, joins, views, triggers, and stored procedures (in multiple languages). It includes most SQL92 and SQL99 data types, including INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, and TIMESTAMP. It also supports storage of binary large objects, including pictures, sounds, or video. It has native programming interfaces for C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, among others, and exceptional documentation.

PostgreSQL boasts sophisticated features such as Multi-Version Concurrency Control (MVCC), point in time recovery, tablespaces, asynchronous replication, nested transactions (savepoints), online/hot backups, a sophisticated query planner/optimizer, and write ahead logging for fault tolerance.

It supports international character sets, multibyte character encodings, Unicode, and it is locale-aware for sorting, case-sensitivity, and formatting. It is highly scalable both in the sheer quantity of data it can manage and in the number of concurrent users it can accommodate. Some general PostgreSQL limits are included in the table below.

Limit	**Value**
Maximum Database Size	Unlimited
Maximum Table Size	32 TB
Maximum Row Size	1.6 TB
Maximum Field Size	1 GB
Maximum Rows per Table	Unlimited
Maximum Columns per Table	250 - 1600 depending on column types
Maximum Indexes per Table	Unlimited

It is obvious that PostgreSQL is more than capable for our data model needs in terms of size. For example, CTI has used PostgreSQL in the CITER project (http://citer.cti.gr) as well, where the database included more than 1100 tables and stored more than 50 printed books, with 1 multimedia DVD and one entire encyclopaedia, for a total of more than 80000 paragraphs translated in 6 languages. PostgreSQL had proved more than adequate to handle this kind of workload.

PostgreSQL also allows the return of partial result sets (with LIMIT/OFFSET) and supports compound, unique, partial, and functional indexes which can use any of its B-tree, R-tree, hash, or GiST storage methods.

Another reason for choosing PostgreSQL over other open source RDBM systems is PostGIS. PostGIS is a project which adds support for geographic objects in PostgreSQL, allowing it to be used as a spatial database for geographic information systems (GIS), much like ESRI's SDE or Oracle's Spatial extension. Since location is an important aspect of TALOS project, we needed a RDBMS with the best possible support for geographic objects.

Another feature of PostgreSQL used in the TALOS project, is the Ltree module that may be used for storing tree-like structures. The tree-like structure of any book or task can be fully visualized with the Ltree datatype available in PostgreSQL, opening new possibilities in structural queries.

Best of all, PostgreSQL's source code is available under the most liberal open source license: the BSD license. This license gives you the freedom to use, modify and distribute PostgreSQL in any form you like, open or closed source.

3 Web Interface

3.1 Introduction

The object of this deliverable is to present the various phases of development of the TALOS web-based interface for the annotation and authoring tool detailed in WP3 of the proposal. The design and implementation of the application is described in detail.

Semantic Data Markup comprises technologies that allow for efficient re-use of existing content and automatic creation of metadata. Data adaptation technology has been developed for maximal re-use of existing content. Spatial and temporal data are important metadata for content used in mobile services. In this WP RACTI has developed automatic parsing technology based on identifying geomarkers in texts and relating them to existing geocoded repositories. Further, using Web scraping technology, Web content is intermixed with authored content to exploit Web engineering and knowledge management expertise. Finally, the metadata has been integrated in an overall metadata framework for structured content.

3.2 System Architecture

The architecture of the system is guided by the ultimate purpose of the component subsystems which are then integrated into a single user-friendly environment which can be readily used by authors.

At the outset the requirements and specification of all the relevant modules and their functionality are defined. The next step requires the design of the interfacing between the various modules that make up the overall authoring system. This ensures that a common communication protocol is used between modules, in order that independent work can be simultaneously carried out between development teams thereby facilitating substitution of modules by functional equivalents.

The system description is best observed from two view-points: the logical viewpoint, which is a synopsis of the abstract specification of all the system modules and their interactions and, secondly, the physical viewpoint which lays out the details of the system architecture including the development and target platforms as well as the software tools that are used.

3.2.1 Logical View of the system

The information flow in response to a user query and the interactions between the modules is shown in the figure below. Through the Graphical User Interface (GUI), the user makes queries which are executed by the Data Management (DM) module. Each query activates a specific subsystem that is related to a specific view of the data stored in the database. The logical view of the system is indicative for the development of the application. The system is totally transparent to the user.

3.2.2 Physical View of the system

The logical system view does not describe how the modules are implemented or how communication between these modules is realized. These details are left to the subsections presented below.

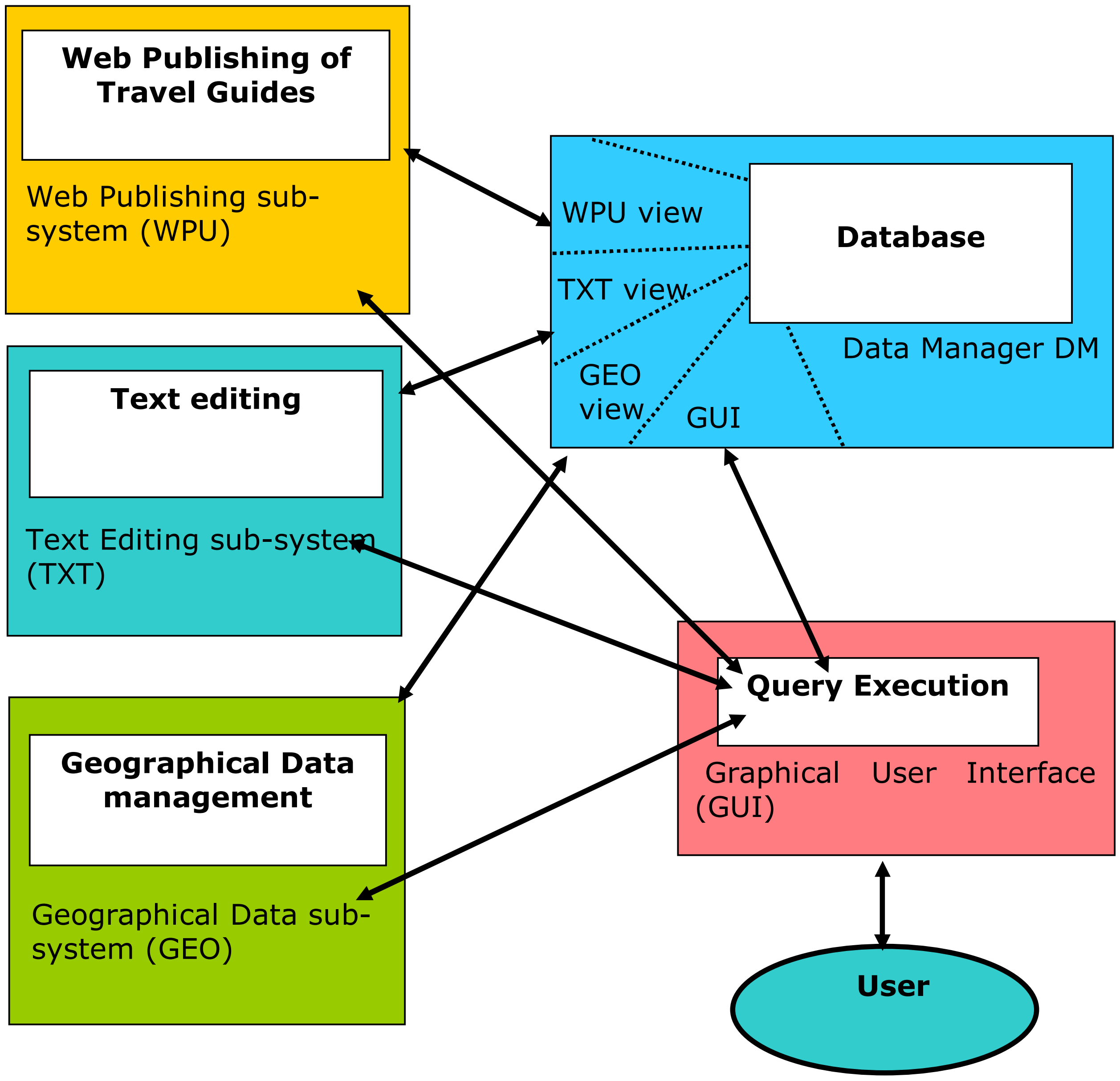

Web Publishing of Travel Guides
Web Publishing sub-system (WPU)
WPU view
Database
TXT view
Data Manager DM
GEO view
GUI
Text editing
Text Editing sub-system (TXT)
Query Execution
Graphical User Interface (GUI)
Geographical Data management
Geographical Data sub-system (GEO)
User

Figure 1 – Interactions among the sub-systems

3.2.3 Development environment

The requirement for processing and accessing of large quantities of information distributed over a wide area using web-based techniques requires specialized software tools with embedded database management as well as efficient and effective communications. Re-usability of code of the software system is imperative for rapid development of web-based applications.

A suitable development environment which satisfies all these requirements is Ruby on Rails (ROR or simply 'Rails'). This environment has been used to develop all the modules of the system.

3.2.4 Brief description of the 'Rails' environment

'Rails' is an open source environment that is extensively used for the development of web-based systems ('Tweeter' is an example) and is written in Ruby, a high level object based language. The end code is far more compact than any other environment and the time to develop far shorter.

The 'Rails' environment continuously evolves in response to user requirements and offers high usability, compactness, robustness and inter-operability. A few of the properties of 'Rails' stand out:

DRY or "Don't Repeat Yourself": 'Rails' supports the principles of DRY programming. Once a change is decided upon its not necessary to modify the code in more than one authoritative location. 'Rails' also adheres to the DRY principle when it comes to implementing cutting edge techniques such as Ajax (Asynchronous JavaScript and XML). Ajax is an approach that allows a web application to replace content in the user's browser dynamically or to exchange form data with the server without reloading a page. Developers often find themselves duplicating code while creating Ajax applications even though browsers that do not support Ajax. 'Rails' makes it easy to treat each browser generation without duplication of code.

Convention Over Configuration refers to the fact that 'Rails' assumes a number of defaults for a typical web application. 'Rails' has been created in such a way that it does not require excessive configuration, if certain standard conventions are followed. The result is that lengthy configuration files are not necessary. Indeed there is no need to change these defaults. 'Rails' requires only a single short configuration file in order to execute an application. This file is used to establish the database connection and supplies 'Rails' with the necessary database server type, server name, user name and password for each environment.

Agile Development: traditional approaches to software development, such as iterative development and the waterfall model, normally attempt to define a long-running and static plan for the goals and needs of an application using predictive methods. These models usually approach applications from the bottom-up by working on the data first.

In contrast, agile development methods use an adaptive approach. Small teams iteratively complete small units of a project. Before starting any iteration, the team re-evaluates the priorities for the subtask, which could have changed in the previous iteration. Agile developers design their applications from the top-down, starting with the design, which may be as simple as an outline of the interface.

When an application is built using Agile methods, it is less likely to veer out of control during the development cycle, due to the ongoing efforts of the team which adjusts priorities. By spending less time creating functional specifications and long-running schedules, developers using Agile methods can jumpstart an application's development.

MVC Architecture

MVC (Model View Controller) is an architecture structure for software applications. An application is broken down into the following three components:

- models, for handling data and operational logic
- controllers, for handling the user interface and application logic
- views, for handling graphical user interface objects and presentation logic. This structure leads to user requests being processed as follows:
- The browser on the client sends a request for a page to the controller on the server.
- The controller retrieves the data it requires from the model in order to respond to the request.
- The controller renders the page and sends it to the view.

- The view sends the page back to the client for the browser to display.
- This process is depicted in Figure 2.

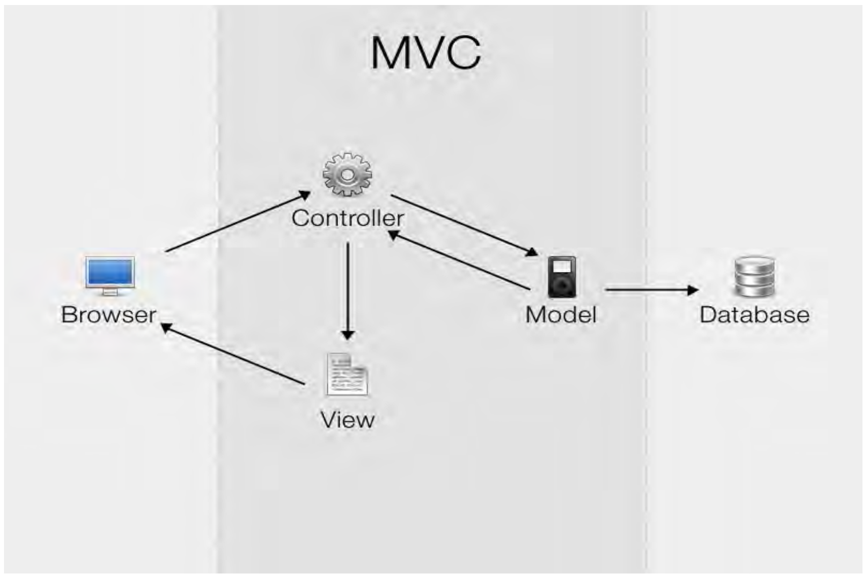


Figure 2 Processing a page request in an MVC architecture

Separating a software system into three components is helpful for the following reasons: it

- improves scalability (the ability for an application to grow): if an application experiences slow performance issues because database access is slow, for example, the solution may most likely lie in the hardware running the database without other components being affected.
- makes maintenance easier because the components have a low dependency on each other. Changes to one (e.g. to change functionality) does not affect the operation of another.
- promotes software reuse by multiple views.
- makes for easy distribution of the application. Separation of code inherently implies that each component could reside on a separate machine.

‘Rails’ implements the concept that models, views, and controllers should be kept separate by storing the code for each of these elements as separate files in separate directories, as shown in Figure 3.

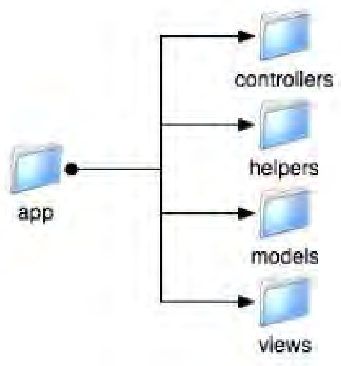


Figure 3. The app(lication) subdirectory

Models
The model represents the application data and the rules for their management. In 'Rails' the models are used primarily for the management of the rules interaction with a table in the database. In most cases each model represents one table of the database. The application logic is based on the various models and their interactions.

Views
Views represent the application interface with the user. In 'Rails', views are mainly HTML files with embedded code in the Ruby language that executes commands for data display. The Views provide all pertinent data to the web browser according to the selections of the user and the application requirements.

Controllers
The controllers connect the models with the views. In 'Rails', the controllers are responsible for processing requests from the web browser, requesting data from the models and transferring this data to the views for their display.

'Rails' Cycle
The following diagram shows the sequence of the responses to each request posed by the user in the 'Rails' Architecture. Request A activates controller A, while request B activates controller B and so on. Each controller call results in a specific response to the request through a route that is selected in the background of the architecture.

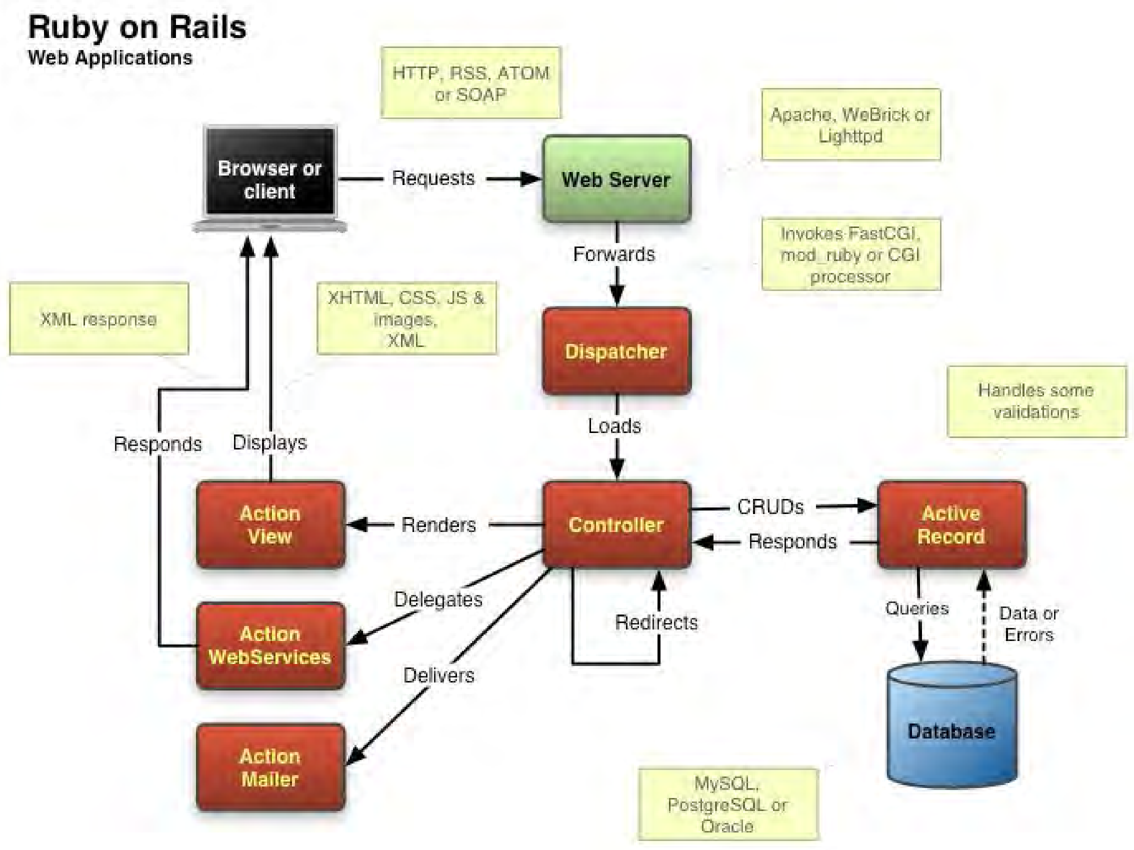


Figure 3. Web Application in the 'Rails' Environment

Principal 'Rails' Modules

The 'Rails' environment includes the following set of modules for web applications:

- `Action Controller`
- `Action View`
- `Active Record`
- `Action Mailer`
- `Active Resource`
- `Railties`
- `Active Support`

The installation and use of the various system modules clearly depends on the requirements of the application. However, three are basic and are described briefly below:

Action Controller

`ActionController` is the component that handles browser requests and facilitates communication between the model and the view. The controllers in the application will inherit from this class. It forms part of the `ActionPack` library, a collection of 'Rails' components.

Action View

`ActionView` is the component that handles the presentation of pages returned to the client. Views inherit from this class, which is also part of the `ActionPack` library.

Active Record

`ActiveRecord` is designed to handle all tasks of an application that relate to the database, including:

- establishing a connection to the database server
- retrieving data from a table
- storing new data in the database

`ActiveRecord` is independent of the database and provides all the basic CRUD (Create, Retrieve, Update, Delete) operations, complex data retrieval operations and has the ability to relate data as it is required in a relational database.

Some additional features of 'Rails'

- The 'Rails' environment supports several databases, MySQL, PostgreSQL, SQLite, SQL Server, DB2, Oracle, thereby providing freedom to the development engineer to select a database appropriate for a specific application.
- The 'Rails' environment creates automatically a complete set of operations (Scaffolding) for CRUD and the respective views of a table in the database.
- Allows extensive use of libraries for JavaScript and Ajax.
- Includes validation mechanisms for checking the correctness of the data and error trapping.
- Provides templates for error detection during the test phase of the various parts of the application and of its entity.
- Requires far less code compared to other development environments, resulting in faster development and code maintenance and future improvements.

Development Platform

The software tools used for the development of the web application include open source or license free software for the operational systems Windows XP/Vista:

Ruby	:	Objective oriented language	Open Source
'Rails'	:	Web-based applications environment	Open Source
Aptana Studio	:	Web-based applications platform with embedded RadRails for the development on 'Rails'	License Free
PostgreSQL	:	Relational Data Base	License Free
GoogleMaps API	:	Application Programming Interface to integrate Google Maps (web mapping service application)	License Free
Javascript	:	Scripting language used to obtain access to	License Free

objects within other applications

3.3 Description of the Web Application

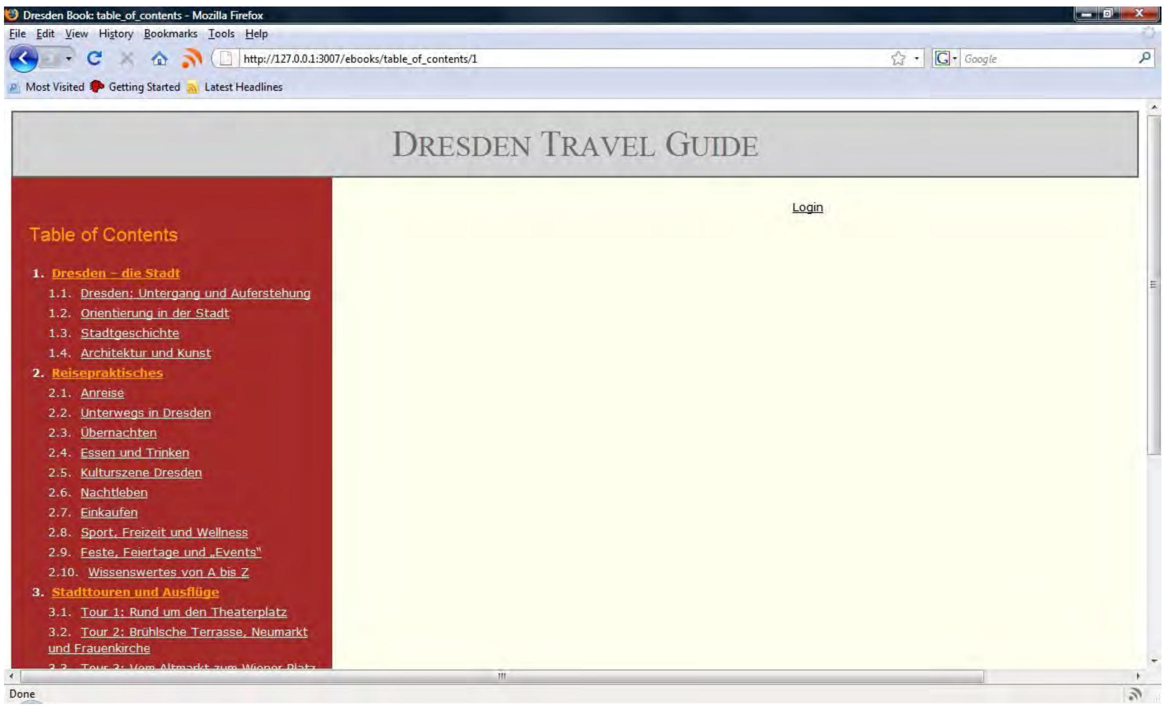
The main parts of the web application are the following:

1. Publishing selected travel guides on the internet
2. Editing text and images of the travel guides
3. Addition of metadata to the text

The application includes a public part where the user can navigate to the complete Travel Guide published or to specific parts and a part of limited access where the user can edit the text and produce new metadata. The two sections are described in detail in the following:

3.4 Public Part

The initial page of the web application is shown below:


Dresden Book: table_of_contents - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1:3007/ebooks/table_of_contents/1
Most Visited Getting Started Latest Headlines
Login
Done

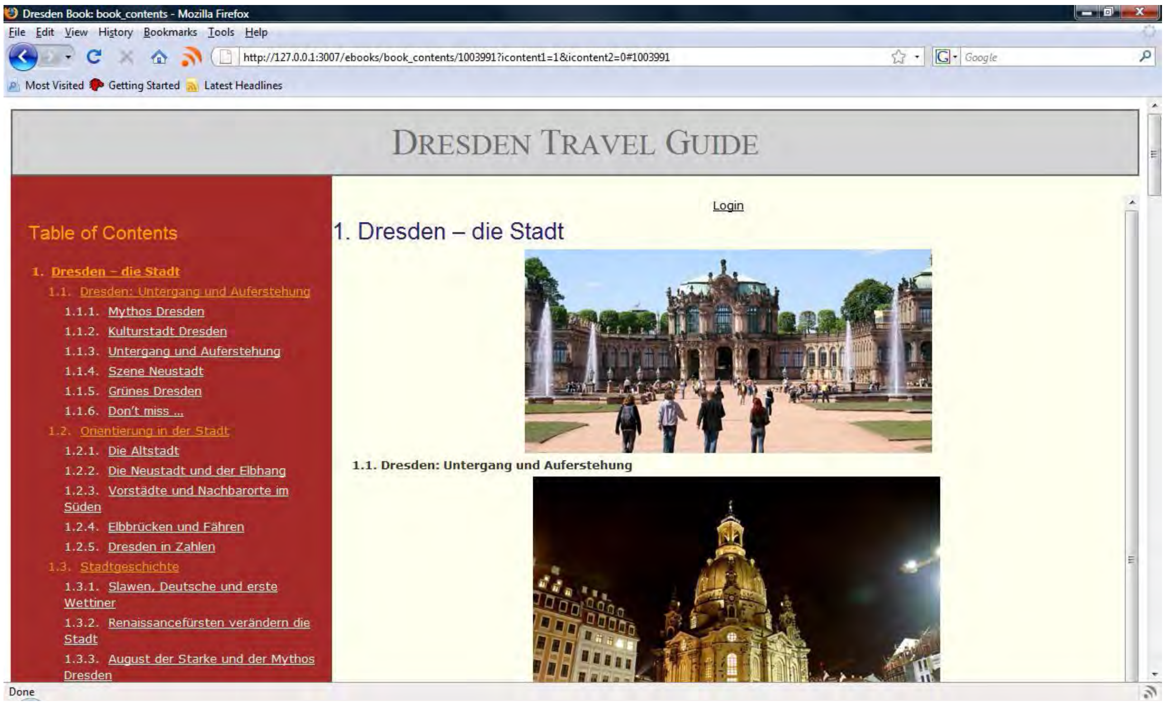
DRESDEN TRAVEL GUIDE

Table of Contents

1. Dresden – die Stadt
   1.1. Dresden: Untergang und Auferstehung
   1.2. Orientierung in der Stadt
   1.3. Stadtgeschichte
   1.4. Architektur und Kunst
2. Reisepraktisches
   2.1. Anreise
   2.2. Unterwegs in Dresden
   2.3. Übernachten
   2.4. Essen und Trinken
   2.5. Kulturszene Dresden
   2.6. Nachtleben
   2.7. Einkaufen
   2.8. Sport, Freizeit und Wellness
   2.9. Feste, Feiertage und „Events"
   2.10. Wissenswertes von A bis Z
3. Stadttouren und Ausflüge
   3.1. Tour 1: Rund um den Theaterplatz
   3.2. Tour 2: Brühlsche Terrasse, Neumarkt und Frauenkirche
   3.3. Tour 3: Vom Altmarkt zum Wiener Platz

Initially the table of contents of the Travel Guide, and refers to the City of Dresden in the pilot case. The table of contents has a hierarchical view typical of one seen in a printed book. For simplicity, only two levels of sections are displayed initially. The user can navigate to the various sections and subsections of the Guide by clicking on the specific title of the Table of Contents.

For example, on selecting the first section, namely *1. Dresden – die Stadt*, the following view is displayed:


Dresden Book: book_contents - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1:3007/ebooks/book_contents/1003991?icontent1=1&icontent2=0#1003991
Most Visited
Getting Started
Latest Headlines
Login
Done

DRESDEN TRAVEL GUIDE

Table of Contents

1. Dresden – die Stadt
   1.1. Dresden: Untergang und Auferstehung
      1.1.1. Mythos Dresden
      1.1.2. Kulturstadt Dresden
      1.1.3. Untergang und Auferstehung
      1.1.4. Szene Neustadt
      1.1.5. Grünes Dresden
      1.1.6. Don't miss ...
   1.2. Orientierung in der Stadt
      1.2.1. Die Altstadt
      1.2.2. Die Neustadt und der Elbhang
      1.2.3. Vorstädte und Nachbarorte im Süden
      1.2.4. Elbbrücken und Fähren
      1.2.5. Dresden in Zahlen
   1.3. Stadtgeschichte
      1.3.1. Slawen, Deutsche und erste Wettiner
      1.3.2. Renaissancefürsten verändern die Stadt
      1.3.3. August der Starke und der Mythos Dresden

1. Dresden – die Stadt

1.1. Dresden: Untergang und Auferstehung

As all the text does not fit on the screen, scrolling is available (at the right hand side of the screen). The user can select subsections down three levels for display. At all times, the selected part is displayed at the top of the screen for ease of reading, as shown in the following figure, where section *1.2.1 Die Altstadt* was selected from the Table of contents.

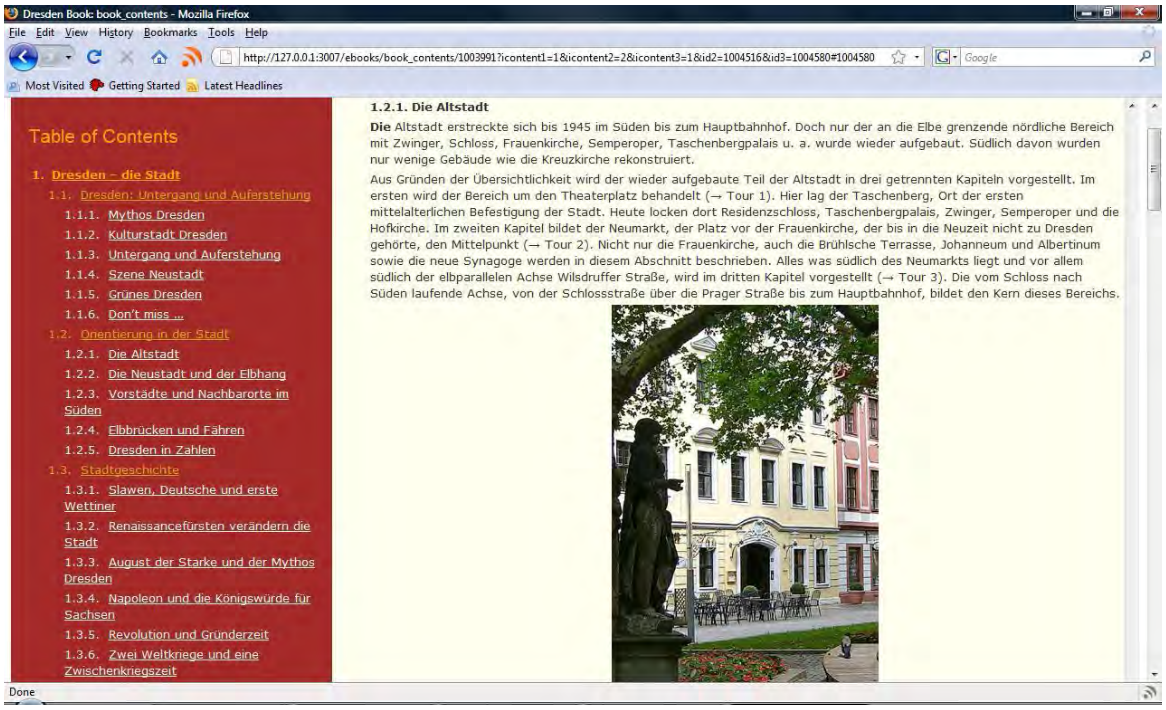

Dresden Book: book_contents - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1:3007/ebooks/book_contents/1003991?icontent1=1&icontent2=2&icontent3=1&id2=1004516&id3=1004580#1004580
Most Visited
Getting Started
Latest Headlines
Done

Table of Contents

1. Dresden – die Stadt
   1.1. Dresden: Untergang und Auferstehung
      1.1.1. Mythos Dresden
      1.1.2. Kulturstadt Dresden
      1.1.3. Untergang und Auferstehung
      1.1.4. Szene Neustadt
      1.1.5. Grünes Dresden
      1.1.6. Don't miss ...
   1.2. Orientierung in der Stadt
      1.2.1. Die Altstadt
      1.2.2. Die Neustadt und der Elbhang
      1.2.3. Vorstädte und Nachbarorte im Süden
      1.2.4. Elbbrücken und Fähren
      1.2.5. Dresden in Zahlen
   1.3. Stadtgeschichte
      1.3.1. Slawen, Deutsche und erste Wettiner
      1.3.2. Renaissancefürsten verändern die Stadt
      1.3.3. August der Starke und der Mythos Dresden
      1.3.4. Napoleon und die Königswürde für Sachsen
      1.3.5. Revolution und Gründerzeit
      1.3.6. Zwei Weltkriege und eine Zwischenkriegszeit

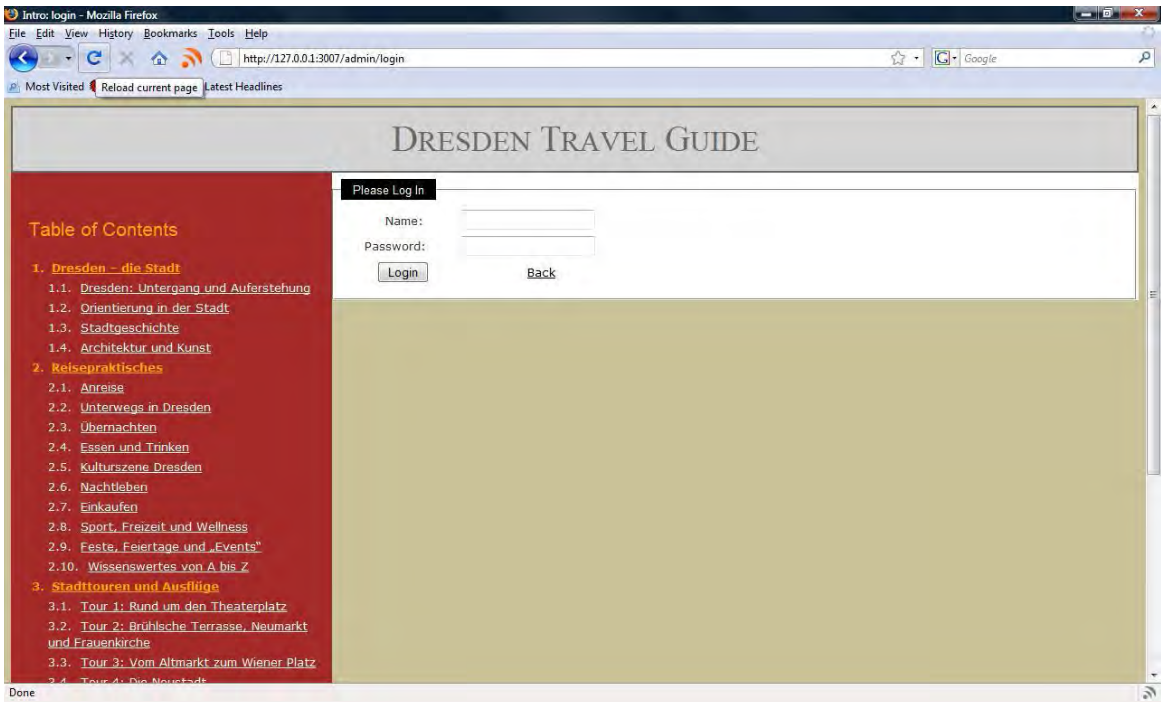
1.2.1. Die Altstadt

Die Altstadt erstreckte sich bis 1945 im Süden bis zum Hauptbahnhof. Doch nur der an die Elbe grenzende nördliche Bereich mit Zwinger, Schloss, Frauenkirche, Semperoper, Taschenbergpalais u. a. wurde wieder aufgebaut. Südlich davon wurden nur wenige Gebäude wie die Kreuzkirche rekonstruiert.

Aus Gründen der Übersichtlichkeit wird der wieder aufgebaute Teil der Altstadt in drei getrennten Kapiteln vorgestellt. Im ersten wird der Bereich um den Theaterplatz behandelt (→ Tour 1). Hier lag der Taschenberg, Ort der ersten mittelalterlichen Befestigung der Stadt. Heute locken dort Residenzschloss, Taschenbergpalais, Zwinger, Semperoper und die Hofkirche. Im zweiten Kapitel bildet der Neumarkt, der Platz vor der Frauenkirche, der bis in die Neuzeit nicht zu Dresden gehörte, den Mittelpunkt (→ Tour 2). Nicht nur die Frauenkirche, auch die Brühlsche Terrasse, Johanneum und Albertinum sowie die neue Synagoge werden in diesem Abschnitt beschrieben. Alles was südlich des Neumarkts liegt und vor allem südlich der elbparallelen Achse Wilsdruffer Straße, wird im dritten Kapitel vorgestellt (→ Tour 3). Die vom Schloss nach Süden laufende Achse, von der Schlossstraße über die Prager Straße bis zum Hauptbahnhof, bildet den Kern dieses Bereichs.

3.5 Limited Access Part

As shown in the previous figures, an authorized user can "Login" at any time to edit the text. Authorized users are the author, the publisher or whoever has the rights to enter this part of the application, provided he/she owns a valid pair of username and password, which is issued by the administrator of the web application. By clicking on the "*Login*" prompt, shown on the top of any public page, the user is requested to enter a valid name and password. Return to the public pages is provided by clicking "*Back*". Levels of access rights can be specified, allowing permission for specific actions, if this is necessary at later stages of development.


Intro: login - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1:3007/admin/login
Google
Most Visited
Reload current page
Latest Headlines
Done

DRESDEN TRAVEL GUIDE

Please Log In

Name:

Password:

Login Back

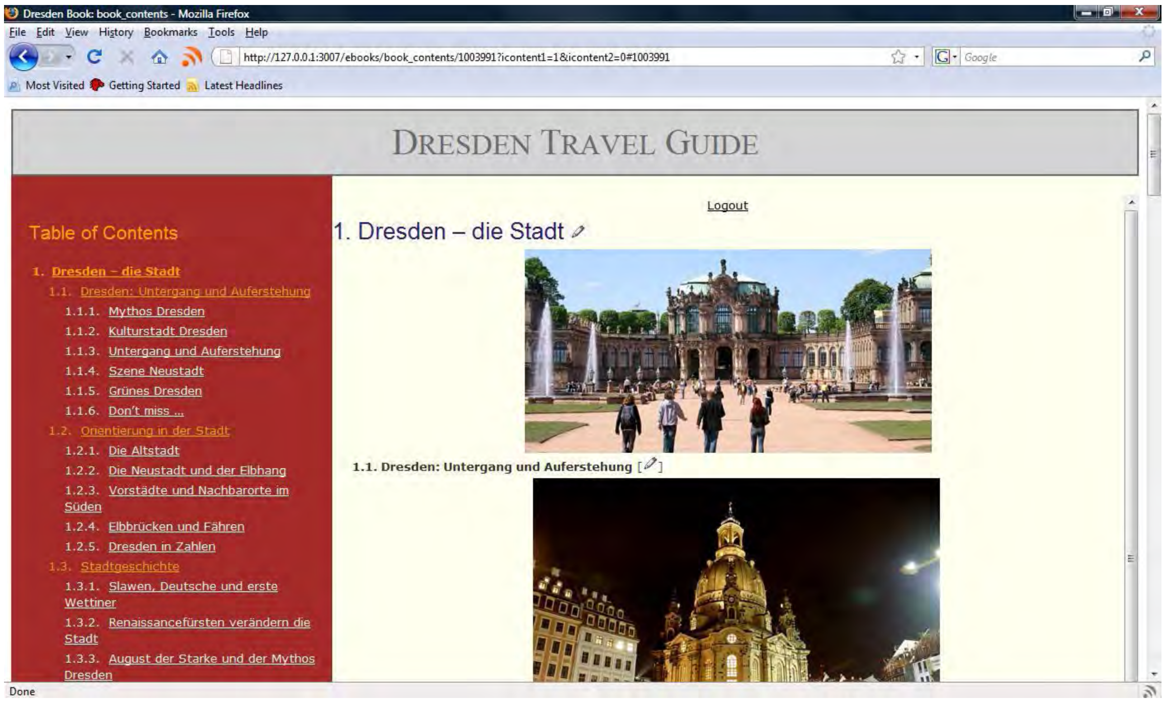
Table of Contents

1. Dresden – die Stadt
   1.1. Dresden: Untergang und Auferstehung
   1.2. Orientierung in der Stadt
   1.3. Stadtgeschichte
   1.4. Architektur und Kunst
2. Reisepraktisches
   2.1. Anreise
   2.2. Unterwegs in Dresden
   2.3. Übernachten
   2.4. Essen und Trinken
   2.5. Kulturszene Dresden
   2.6. Nachtleben
   2.7. Einkaufen
   2.8. Sport, Freizeit und Wellness
   2.9. Feste, Feiertage und „Events"
   2.10. Wissenswertes von A bis Z
3. Stadttouren und Ausflüge
   3.1. Tour 1: Rund um den Theaterplatz
   3.2. Tour 2: Brühlsche Terrasse, Neumarkt und Frauenkirche
   3.3. Tour 3: Vom Altmarkt zum Wiener Platz
   3.4. Tour 4: Die Neustadt

3.5.1 Editing

By clicking on the "*Login*" button, the user enters the part of the application where editing is allowed. A similar window is displayed as before, where navigation in the Travel Guide is driven through the Table of contents, however, next to each section title (down to level three) of the text displayed, an icon to edit [ ] is displayed, as shown in the figure that follows.

It is noted that at any time, the user can log-off by simply clicking on the "*Logout*" prompt displayed at the top of each page.


Dresden Book: book_contents - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1:3007/ebooks/book_contents/1003991?icontent1=1&icontent2=0#1003991
Most Visited Getting Started Latest Headlines
Logout
Done

DRESDEN TRAVEL GUIDE

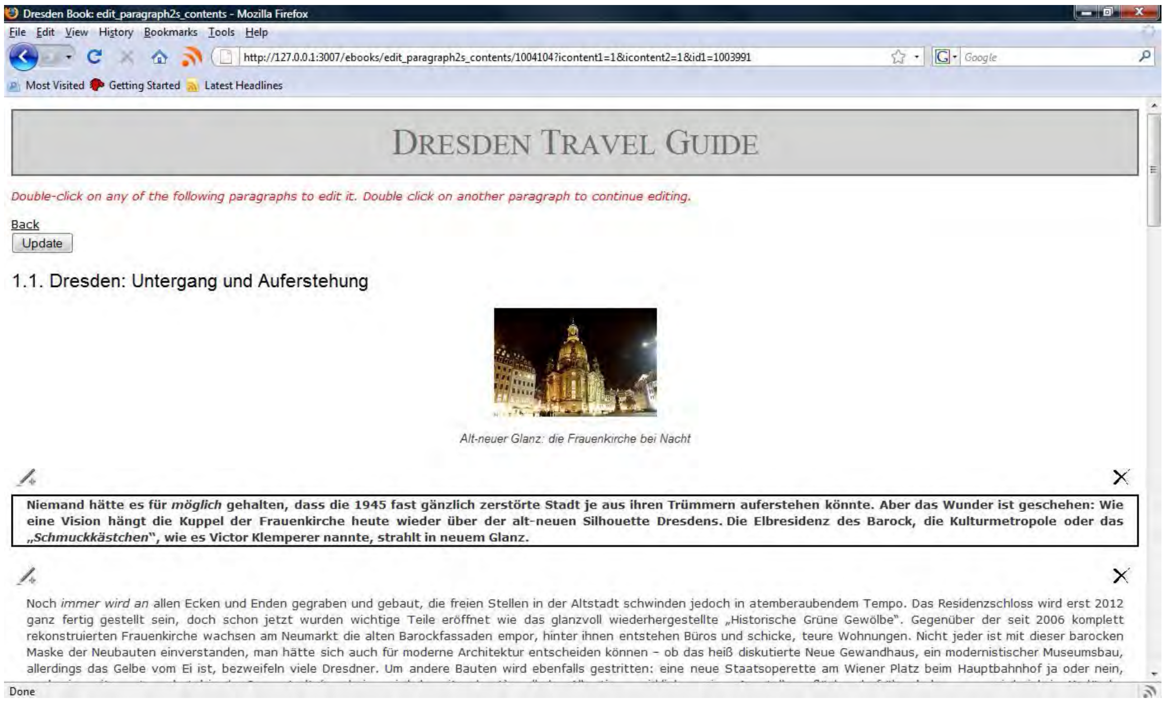
Table of Contents

1. Dresden – die Stadt
   1.1. Dresden: Untergang und Auferstehung
      1.1.1. Mythos Dresden
      1.1.2. Kulturstadt Dresden
      1.1.3. Untergang und Auferstehung
      1.1.4. Szene Neustadt
      1.1.5. Grünes Dresden
      1.1.6. Don't miss ...
   1.2. Orientierung in der Stadt
      1.2.1. Die Altstadt
      1.2.2. Die Neustadt und der Elbhang
      1.2.3. Vorstädte und Nachbarorte im Süden
      1.2.4. Elbbrücken und Fähren
      1.2.5. Dresden in Zahlen
   1.3. Stadtgeschichte
      1.3.1. Slawen, Deutsche und erste Wettiner
      1.3.2. Renaissancefürsten verändern die Stadt
      1.3.3. August der Starke und der Mythos Dresden

1. Dresden – die Stadt

1.1. Dresden: Untergang und Auferstehung

If, for example, some editing is required in section 1.1, the following window is displayed, when the edit icon is clicked:


Dresden Book: edit_paragraph2s_contents - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1:3007/ebooks/edit_paragraph2s_contents/1004104?icontent1=1&icontent2=1&id1=1003991
Most Visited Getting Started Latest Headlines
Back
Update
Done

DRESDEN TRAVEL GUIDE

Double-click on any of the following paragraphs to edit it. Double click on another paragraph to continue editing.

1.1. Dresden: Untergang und Auferstehung

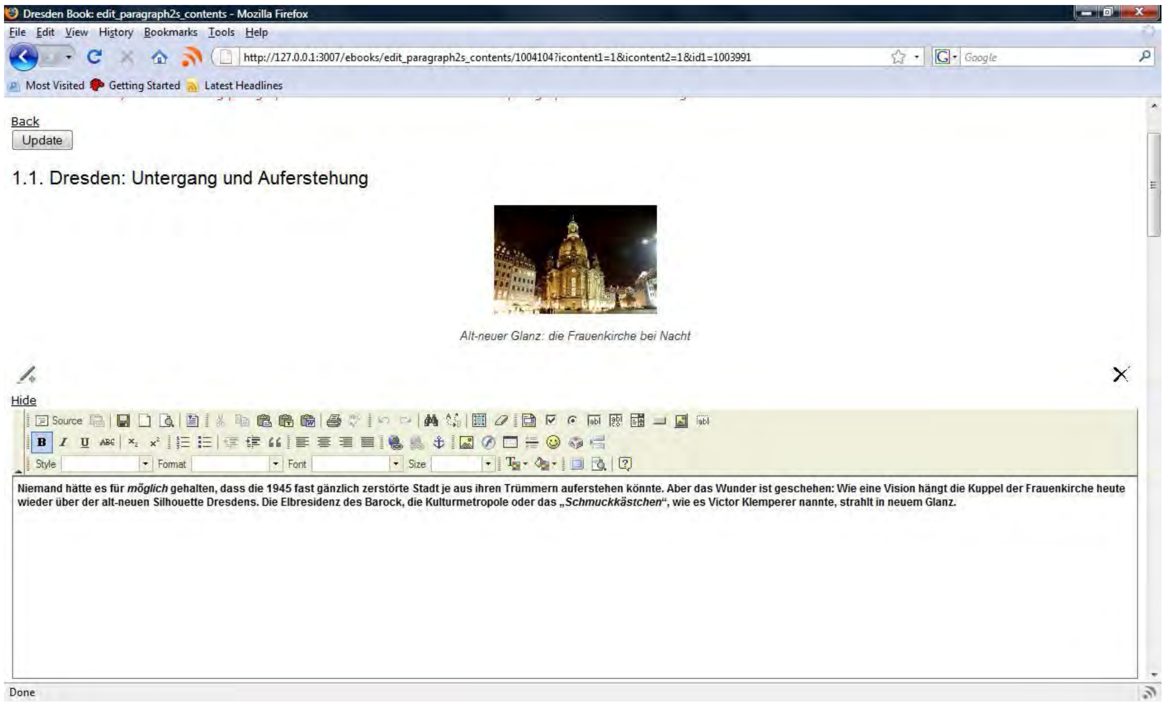
Alt-neuer Glanz: die Frauenkirche bei Nacht

Niemand hätte es für *möglich* gehalten, dass die 1945 fast gänzlich zerstörte Stadt je aus ihren Trümmern auferstehen könnte. Aber das Wunder ist geschehen: Wie eine Vision hängt die Kuppel der Frauenkirche heute wieder über der alt-neuen Silhouette Dresdens. Die Elbresidenz des Barock, die Kulturmetropole oder das *„Schmuckkästchen"*, wie es Victor Klemperer nannte, strahlt in neuem Glanz.

Noch *immer wird an* allen Ecken und Enden gegraben und gebaut, die freien Stellen in der Altstadt schwinden jedoch in atemberaubendem Tempo. Das Residenzschloss wird erst 2012 ganz fertig gestellt sein, doch schon jetzt wurden wichtige Teile eröffnet wie das glanzvoll wiederhergestellte „Historische Grüne Gewölbe". Gegenüber der seit 2006 komplett rekonstruierten Frauenkirche wachsen am Neumarkt die alten Barockfassaden empor, hinter ihnen entstehen Büros und schicke, teure Wohnungen. Nicht jeder ist mit dieser barocken Maske der Neubauten einverstanden, man hätte sich auch für moderne Architektur entscheiden können – ob das heiß diskutierte Neue Gewandhaus, ein modernistischer Museumsbau, allerdings das Gelbe vom Ei ist, bezweifeln viele Dresdner. Um andere Bauten wird ebenfalls gestritten: eine neue Staatsoperette am Wiener Platz beim Hauptbahnhof ja oder nein,

At the top of the page the following instruction is displayed: *Double-click on any of the following paragraphs to edit, is displayed in red. Double click on another paragraph to continue editing*.

When the mouse is on a certain paragraph, this paragraph is displayed within a border as the one selected. By double-clicking on this paragraph and the editor is displayed with functionalities similar to MS Word.


Dresden Book: edit_paragraph2s_contents - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1:3007/ebooks/edit_paragraph2s_contents/1004104?icontent1=1&icontent2=1&id1=1003991
Most Visited Getting Started Latest Headlines
Back
Update
Hide
Source
Style Format Font Size
Done

1.1. Dresden: Untergang und Auferstehung

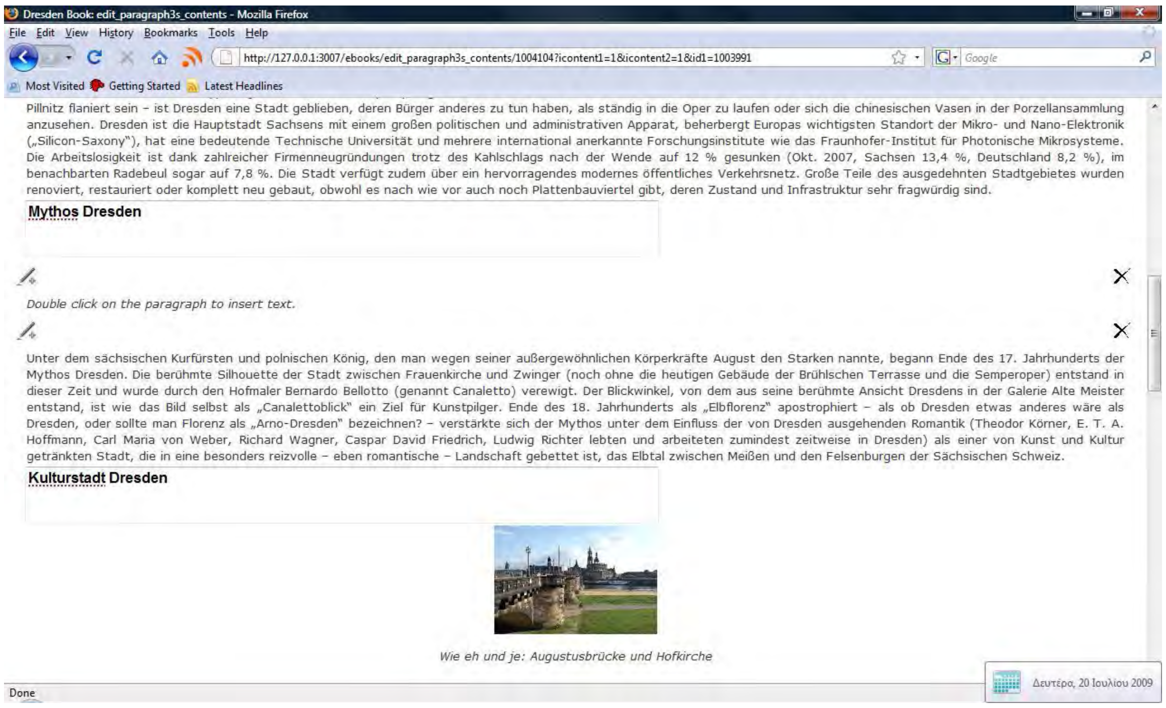
Alt-neuer Glanz: die Frauenkirche bei Nacht

Niemand hätte es für *möglich* gehalten, dass die 1945 fast gänzlich zerstörte Stadt je aus ihren Trümmern auferstehen könnte. Aber das Wunder ist geschehen: Wie eine Vision hängt die Kuppel der Frauenkirche heute wieder über der alt-neuen Silhouette Dresdens. Die Elbresidenz des Barock, die Kulturmetropole oder das „*Schmuckkästchen*", wie es Victor Klemperer nannte, strahlt in neuem Glanz.

Editing is straightforward and when a paragraph is completed, the user can continue by double-clicking on another one. The editor disappears for the previous paragraph and the changes are now displayed, while a new editor opens for the new paragraph selected.

When the user no wishes to continue editing, the editor window will disappear if the "*Hide*" prompt at the top of the editor window is clicked.

At the top of each paragraph there are two icons, one on the left and the other on the right side. The first icon, [ ] is used to insert a paragraph between two contiguous ones or to add a new paragraph at the top or bottom of the specific section. When the user clicks on the Insert icon, an empty paragraph is created prompting the user to "*Double click on the paragraph to insert text*". This action will make the editor window appear for editing, as described above.


Dresden Book: edit_paragraph3s_contents - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://127.0.0.1:3007/ebooks/edit_paragraph3s_contents/1004104?icontent1=1&icontent2=1&id1=1003991
Most Visited Getting Started Latest Headlines
Done
Δευτέρα, 20 Ιουλίου 2009

Pillnitz flaniert sein – ist Dresden eine Stadt geblieben, deren Bürger anderes zu tun haben, als ständig in die Oper zu laufen oder sich die chinesischen Vasen in der Porzellansammlung anzusehen. Dresden ist die Hauptstadt Sachsens mit einem großen politischen und administrativen Apparat, beherbergt Europas wichtigsten Standort der Mikro- und Nano-Elektronik („Silicon-Saxony"), hat eine bedeutende Technische Universität und mehrere international anerkannte Forschungsinstitute wie das Fraunhofer-Institut für Photonische Mikrosysteme. Die Arbeitslosigkeit ist dank zahlreicher Firmenneugründungen trotz des Kahlschlags nach der Wende auf 12 % gesunken (Okt. 2007, Sachsen 13,4 %, Deutschland 8,2 %), im benachbarten Radebeul sogar auf 7,8 %. Die Stadt verfügt zudem über ein hervorragendes modernes öffentliches Verkehrsnetz. Große Teile des ausgedehnten Stadtgebietes wurden renoviert, restauriert oder komplett neu gebaut, obwohl es nach wie vor auch noch Plattenbauviertel gibt, deren Zustand und Infrastruktur sehr fragwürdig sind.

Mythos Dresden

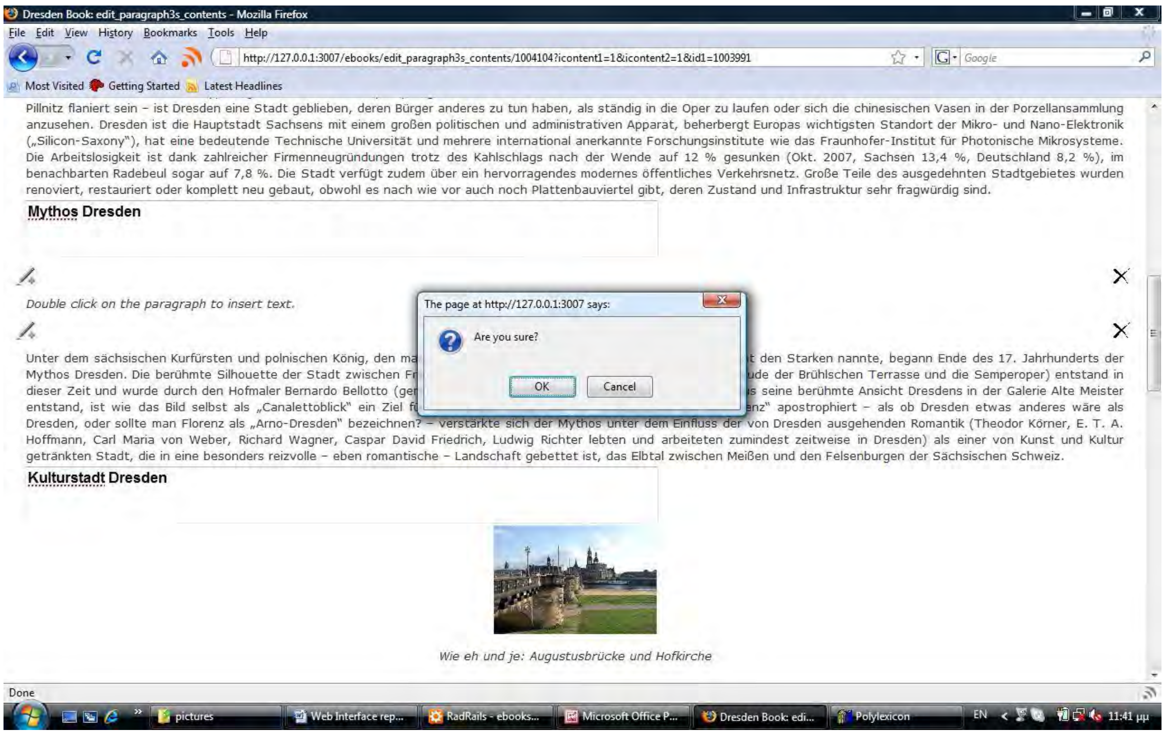
Double click on the paragraph to insert text.

Unter dem sächsischen Kurfürsten und polnischen König, den man wegen seiner außergewöhnlichen Körperkräfte August den Starken nannte, begann Ende des 17. Jahrhunderts der Mythos Dresden. Die berühmte Silhouette der Stadt zwischen Frauenkirche und Zwinger (noch ohne die heutigen Gebäude der Brühlschen Terrasse und die Semperoper) entstand in dieser Zeit und wurde durch den Hofmaler Bernardo Bellotto (genannt Canaletto) verewigt. Der Blickwinkel, von dem aus seine berühmte Ansicht Dresdens in der Galerie Alte Meister entstand, ist wie das Bild selbst als „Canalettoblick" ein Ziel für Kunstpilger. Ende des 18. Jahrhunderts als „Elbflorenz" apostrophiert – als ob Dresden etwas anderes wäre als Dresden, oder sollte man Florenz als „Arno-Dresden" bezeichnen? – verstärkte sich der Mythos unter dem Einfluss der von Dresden ausgehenden Romantik (Theodor Körner, E. T. A. Hoffmann, Carl Maria von Weber, Richard Wagner, Caspar David Friedrich, Ludwig Richter lebten und arbeiteten zumindest zeitweise in Dresden) als einer von Kunst und Kultur getränkten Stadt, die in eine besonders reizvolle – eben romantische – Landschaft gebettet ist, das Elbtal zwischen Meißen und den Felsenburgen der Sächsischen Schweiz.

Kulturstadt Dresden

Wie eh und je: Augustusbrücke und Hofkirche

The second icon [✕] is for deleting the paragraph just below of the icon. A confirmation is required:



Dresden Book: edit_paragraph3s_contents - Mozilla Firefox

http://127.0.0.1:3007/ebooks/edit_paragraph3s_contents/1004104?icontent1=1&icontent2=1&id1=1003991

Pillnitz flaniert sein – ist Dresden eine Stadt geblieben, deren Bürger anderes zu tun haben, als ständig in die Oper zu laufen oder sich die chinesischen Vasen in der Porzellansammlung anzusehen. Dresden ist die Hauptstadt Sachsens mit einem großen politischen und administrativen Apparat, beherbergt Europas wichtigsten Standort der Mikro- und Nano-Elektronik („Silicon-Saxony"), hat eine bedeutende Technische Universität und mehrere international anerkannte Forschungsinstitute wie das Fraunhofer-Institut für Photonische Mikrosysteme. Die Arbeitslosigkeit ist dank zahlreicher Firmenneugründungen trotz des Kahlschlags nach der Wende auf 12 % gesunken (Okt. 2007, Sachsen 13,4 %, Deutschland 8,2 %), im benachbarten Radebeul sogar auf 7,8 %. Die Stadt verfügt zudem über ein hervorragendes modernes öffentliches Verkehrsnetz. Große Teile des ausgedehnten Stadtgebietes wurden renoviert, restauriert oder komplett neu gebaut, obwohl es nach wie vor auch noch Plattenbauviertel gibt, deren Zustand und Infrastruktur sehr fragwürdig sind.

Mythos Dresden

Double click on the paragraph to insert text.

The page at http://127.0.0.1:3007 says:

Are you sure?

Unter dem sächsischen Kurfürsten und polnischen König, den ma … t den Starken nannte, begann Ende des 17. Jahrhunderts der Mythos Dresden. Die berühmte Silhouette der Stadt zwischen Fr … ude der Brühlschen Terrasse und die Semperoper) entstand in dieser Zeit und wurde durch den Hofmaler Bernardo Bellotto (ger … s seine berühmte Ansicht Dresdens in der Galerie Alte Meister entstand, ist wie das Bild selbst als „Canalettoblick" ein Ziel fü … enz" apostrophiert – als ob Dresden etwas anderes wäre als Dresden, oder sollte man Florenz als „Arno-Dresden" bezeichnen? – verstärkte sich der Mythos unter dem Einfluss der von Dresden ausgehenden Romantik (Theodor Körner, E. T. A. Hoffmann, Carl Maria von Weber, Richard Wagner, Caspar David Friedrich, Ludwig Richter lebten und arbeiteten zumindest zeitweise in Dresden) als einer von Kunst und Kultur getränkten Stadt, die in eine besonders reizvolle – eben romantische – Landschaft gebettet ist, das Elbtal zwischen Meißen und den Felsenburgen der Sächsischen Schweiz.

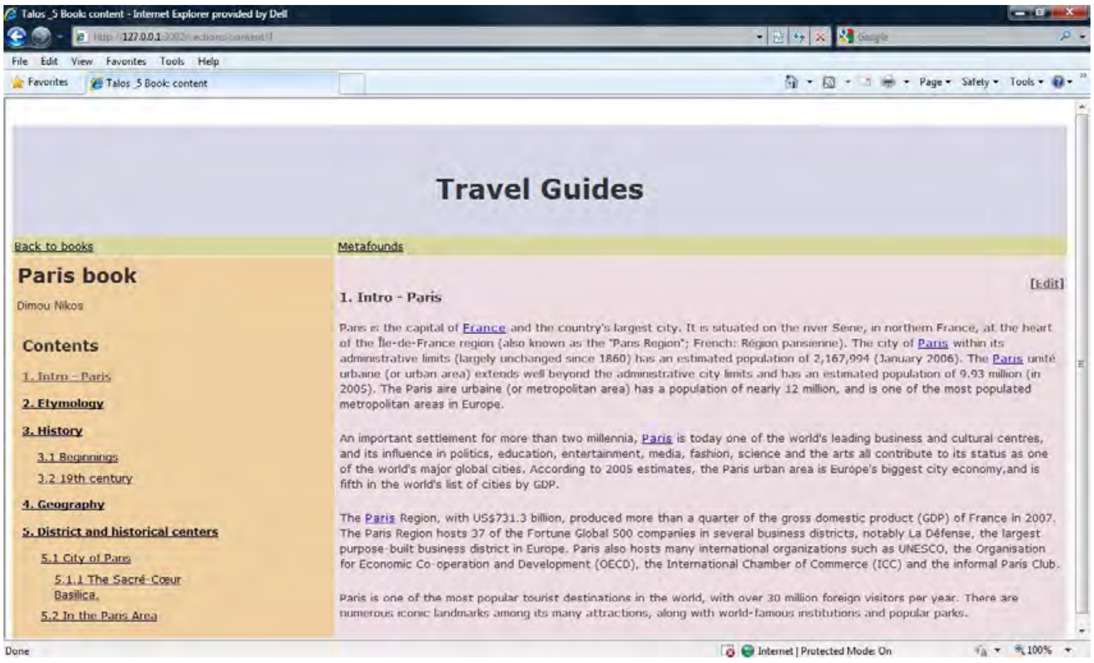
Kulturstadt Dresden

Wie eh und je: Augustusbrücke und Hofkirche

Finally, all changes are stored in the database when the user clicks on the "*Update*" button. Updating the text can be used as many times as the user wishes. The window for editing will remain active while the user is transferred to the published text by clicking on "*Back*". These two options are displayed at the top and bottom of the page, for convenience.

3.5.2 Metadata

An authorized user can add links to geographic information in the text which is stored in the database which becomes available when the links are clicked. Such links are displayed in the following figure.



Talos_5 Book: content - Internet Explorer provided by Dell

Travel Guides

Back to books

Metafounds

Paris book

Dimou Nikos

Contents

1. Intro - Paris
2. Etymology
3. History
   - 3.1 Beginnings
   - 3.2 19th century
4. Geography
5. District and historical centers
   - 5.1 City of Paris
     - 5.1.1 The Sacré-Cœur Basilica.
   - 5.2 In the Paris Area

[Edit]

1. Intro - Paris

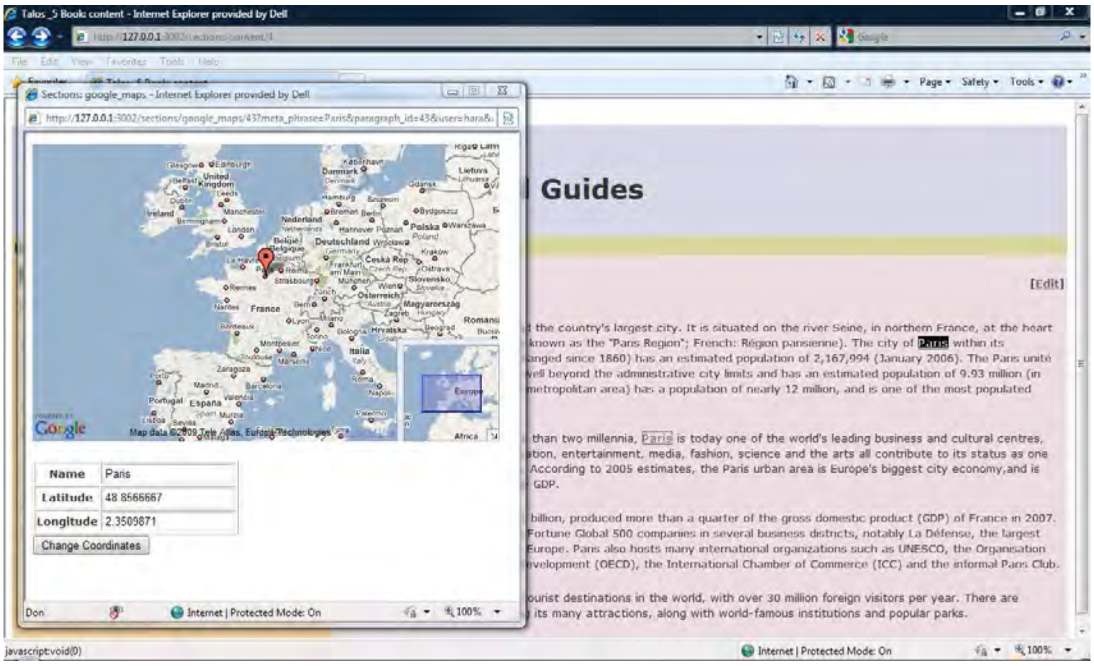
Paris is the capital of France and the country's largest city. It is situated on the river Seine, in northern France, at the heart of the Île-de-France region (also known as the "Paris Region"; French: Région parisienne). The city of Paris within its administrative limits (largely unchanged since 1860) has an estimated population of 2,167,994 (January 2006). The Paris unité urbaine (or urban area) extends well beyond the administrative city limits and has an estimated population of 9.93 million (in 2005). The Paris aire urbaine (or metropolitan area) has a population of nearly 12 million, and is one of the most populated metropolitan areas in Europe.

An important settlement for more than two millennia, Paris is today one of the world's leading business and cultural centres, and its influence in politics, education, entertainment, media, fashion, science and the arts all contribute to its status as one of the world's major global cities. According to 2005 estimates, the Paris urban area is Europe's biggest city economy,and is fifth in the world's list of cities by GDP.

The Paris Region, with US$731.3 billion, produced more than a quarter of the gross domestic product (GDP) of France in 2007. The Paris Region hosts 37 of the Fortune Global 500 companies in several business districts, notably La Défense, the largest purpose-built business district in Europe. Paris also hosts many international organizations such as UNESCO, the Organisation for Economic Co-operation and Development (OECD), the International Chamber of Commerce (ICC) and the informal Paris Club.

Paris is one of the most popular tourist destinations in the world, with over 30 million foreign visitors per year. There are numerous iconic landmarks among its many attractions, along with world-famous institutions and popular parks.

By clicking on "Paris", for example, a window with the geographic coordinates of the place is displayed:



Sections: google_maps - Internet Explorer provided by Dell

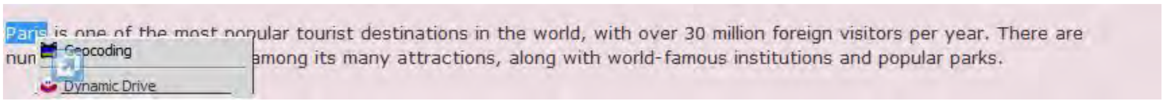
http://127.0.0.1:3002/sections/google_maps/43?meta_phrase=Paris&paragraph_id=43&user=hara&

Name	Paris
Latitude	48.8566667
Longitude	2.3509871

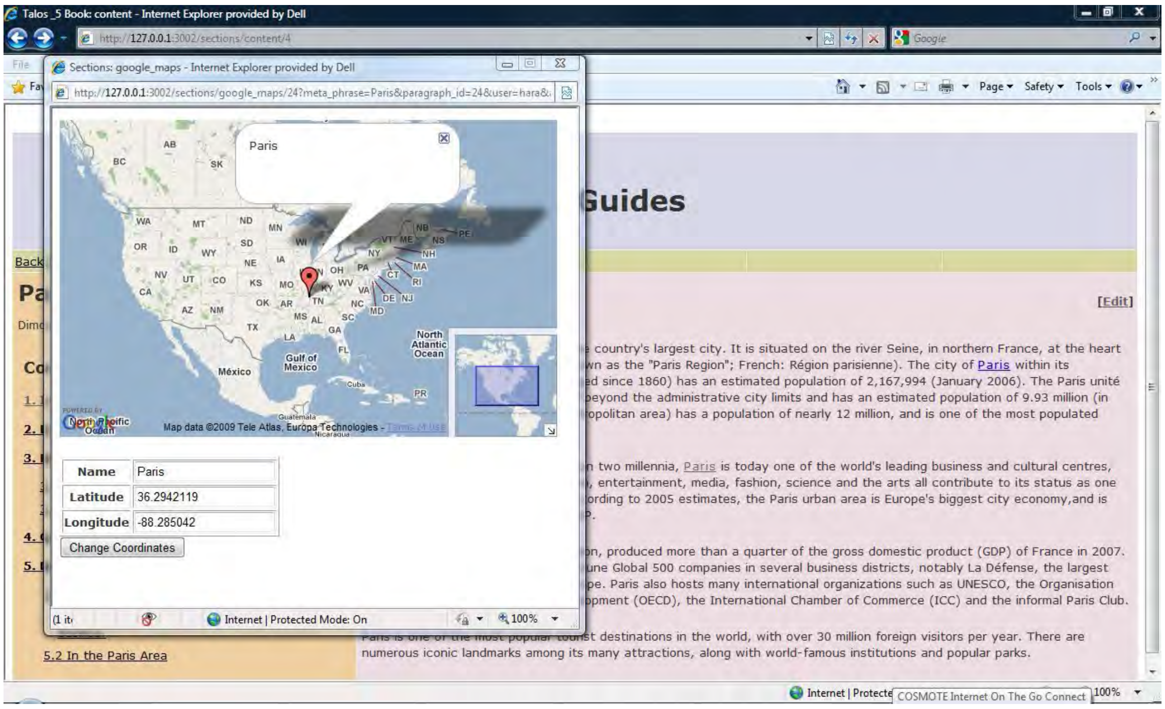
Change Coordinates

The user can change the coordinates, if necessary by clicking on the button "*Change Coordinates*". This option is available not only in all stored links, but also when new geographical data is added, as it is shown in the next window.

To enter a new geographical data, the user must select the text by keeping the mouse pressed down and dragging it on the text, "Paris", in this case. With the text selected and a right click, a pop-up window will show up prompting for "*Geo-coding*":


Paris is one of the most popular tourist destinations in the world, with over 30 million foreign visitors per year. There are numerous iconic landmarks among its many attractions, along with world-famous institutions and popular parks.

On selecting this, a new window appears with Geographical Coordinates of "Paris" provided by Google Maps:


Talos _5 Book: content - Internet Explorer provided by Dell
http://127.0.0.1:3002/sections/content/4
Sections: google_maps - Internet Explorer provided by Dell
http://127.0.0.1:3002/sections/google_maps/24?meta_phrase=Paris&paragraph_id=24&user=hara&

Name	Paris
Latitude	36.2942119
Longitude	-88.285042

Change Coordinates

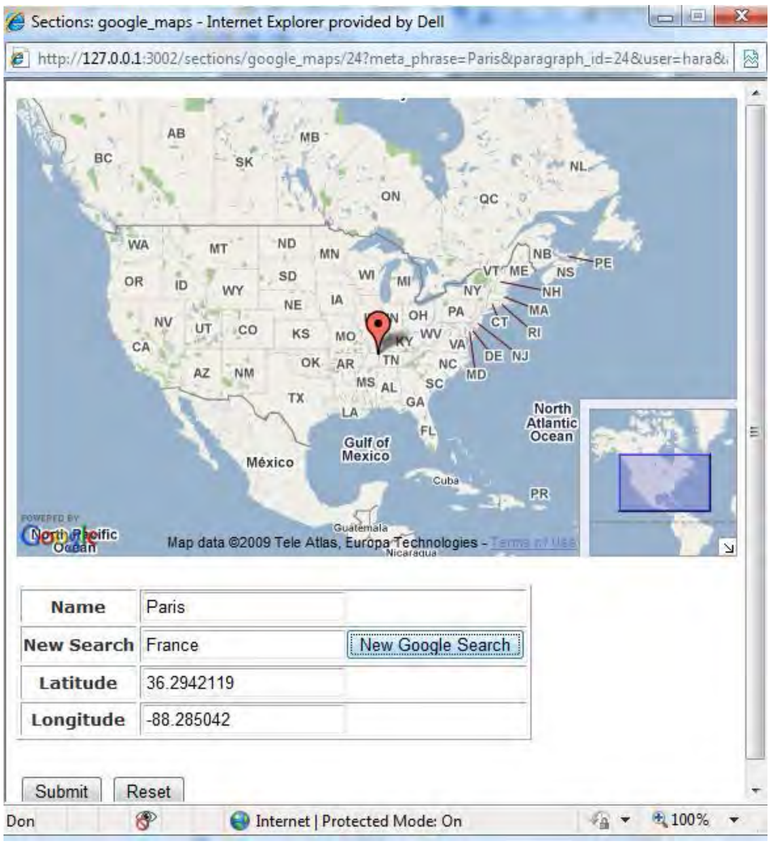
country's largest city. It is situated on the river Seine, in northern France, at the heart wn as the "Paris Region"; French: Région parisienne). The city of Paris within its ed since 1860) has an estimated population of 2,167,994 (January 2006). The Paris unité beyond the administrative city limits and has an estimated population of 9.93 million (in opolitan area) has a population of nearly 12 million, and is one of the most populated

n two millennia, Paris is today one of the world's leading business and cultural centres, , entertainment, media, fashion, science and the arts all contribute to its status as one ording to 2005 estimates, the Paris urban area is Europe's biggest city economy,and is

on, produced more than a quarter of the gross domestic product (GDP) of France in 2007. une Global 500 companies in several business districts, notably La Défense, the largest pe. Paris also hosts many international organizations such as UNESCO, the Organisation pment (OECD), the International Chamber of Commerce (ICC) and the informal Paris Club.

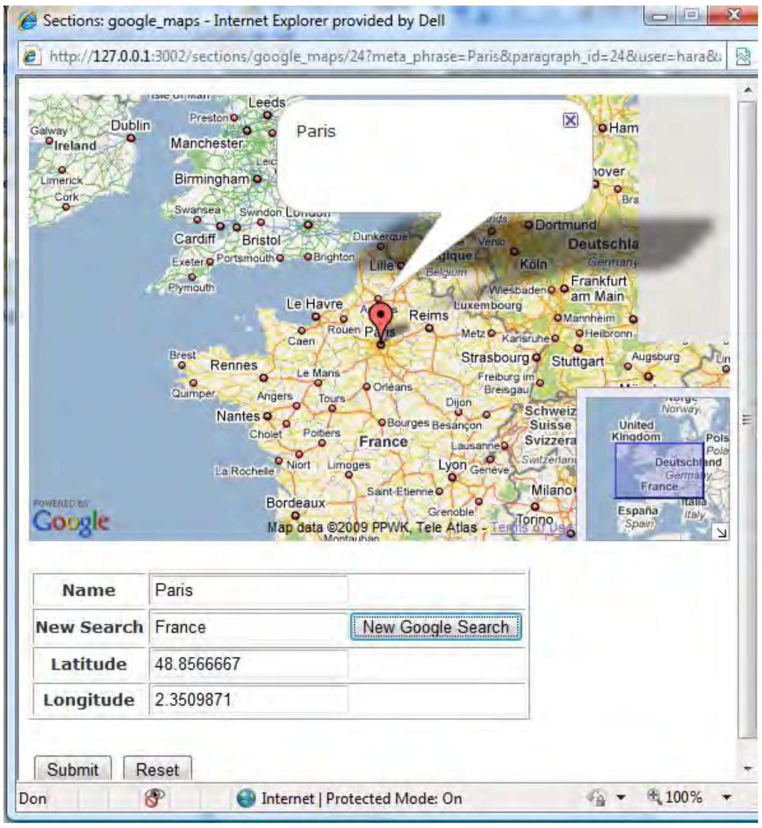
Paris is one of the most popular tourist destinations in the world, with over 30 million foreign visitors per year. There are numerous iconic landmarks among its many attractions, along with world-famous institutions and popular parks.
Paris
Guides
[Edit]
5.2 In the Paris Area
Map data ©2009 Tele Atlas, Europa Technologies
Internet | Protected Mode: On
100%
Page
Safety
Tools
COSMOTE Internet On The Go Connect

As the coordinates found by Google Maps may be not the correct ones, the user can change them by clicking on the "*Change Coordinates*" button:


Sections: google_maps - Internet Explorer provided by Dell
http://127.0.0.1:3002/sections/google_maps/24?meta_phrase=Paris&paragraph_id=24&user=hara&
Map data ©2009 Tele Atlas, Europa Technologies - Terms of Use
Submit
Reset
Internet | Protected Mode: On
100%

Name	Paris	
New Search	France	New Google Search
Latitude	36.2942119	
Longitude	-88.285042	

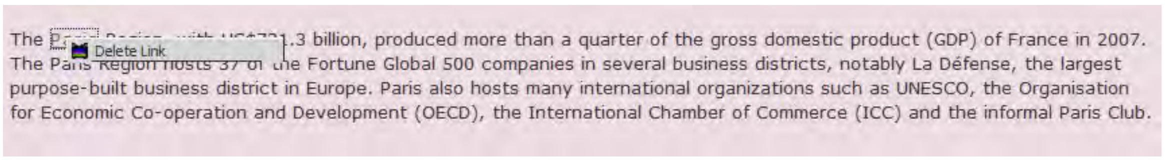
More data can be inserted for the "*New Search*", so when "France" is typed, a "*New Google Search*" request is executed.


Sections: google_maps - Internet Explorer provided by Dell
http://127.0.0.1:3002/sections/google_maps/24?meta_phrase=Paris&paragraph_id=24&user=hara&
Paris
Map data ©2009 PPWK, Tele Atlas - Terms of Use
Submit
Reset
Internet | Protected Mode: On
100%

Name	Paris	
New Search	France	New Google Search
Latitude	48.8566667	
Longitude	2.3509871	

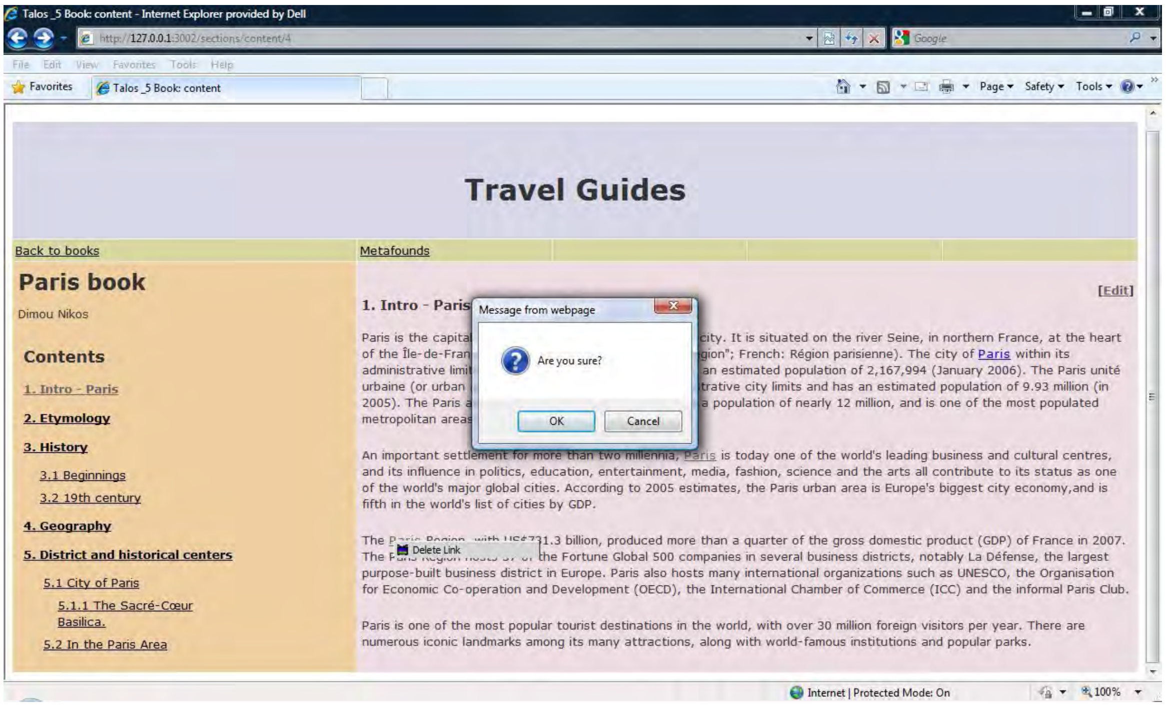
If these are the correct coordinates, then the user concurs by clicking on the "*Submit*" Button, and this information is stored in the database for further use. The "*Reset*" button brings back the original coordinates, and if no action is desired, the user can simply close the Google Maps window and return to the travel guide.

To delete geographical data, a right click on the link prompts for deletion, as shown below:



The Paris Region with US$731.3 billion, produced more than a quarter of the gross domestic product (GDP) of France in 2007. The Paris Region hosts 37 of the Fortune Global 500 companies in several business districts, notably La Défense, the largest purpose-built business district in Europe. Paris also hosts many international organizations such as UNESCO, the Organisation for Economic Co-operation and Development (OECD), the International Chamber of Commerce (ICC) and the informal Paris Club.

The data is deleted, following user confirmation:


Talos _5 Book: content - Internet Explorer provided by Dell
http://127.0.0.1:3002/sections/content/4
Favorites
Talos _5 Book: content
Back to books
Metafounds
[Edit]
Message from webpage
OK
Cancel
Delete Link
Internet | Protected Mode: On
100%

Travel Guides

Paris book

Dimou Nikos

Contents

1. Intro - Paris
2. Etymology
3. History
3.1 Beginnings
3.2 19th century
4. Geography
5. District and historical centers
5.1 City of Paris
5.1.1 The Sacré-Cœur Basilica.
5.2 In the Paris Area

1. Intro - Paris

Are you sure?

Paris is the capital ... city. It is situated on the river Seine, in northern France, at the heart of the Île-de-Fran... gion"; French: Région parisienne). The city of Paris within its administrative limit... an estimated population of 2,167,994 (January 2006). The Paris unité urbaine (or urban ... trative city limits and has an estimated population of 9.93 million (in 2005). The Paris a... a population of nearly 12 million, and is one of the most populated metropolitan areas

An important settlement for more than two millennia, Paris is today one of the world's leading business and cultural centres, and its influence in politics, education, entertainment, media, fashion, science and the arts all contribute to its status as one of the world's major global cities. According to 2005 estimates, the Paris urban area is Europe's biggest city economy,and is fifth in the world's list of cities by GDP.

The Paris Region with US$731.3 billion, produced more than a quarter of the gross domestic product (GDP) of France in 2007. The Paris Region hosts 37 of the Fortune Global 500 companies in several business districts, notably La Défense, the largest purpose-built business district in Europe. Paris also hosts many international organizations such as UNESCO, the Organisation for Economic Co-operation and Development (OECD), the International Chamber of Commerce (ICC) and the informal Paris Club.

Paris is one of the most popular tourist destinations in the world, with over 30 million foreign visitors per year. There are numerous iconic landmarks among its many attractions, along with world-famous institutions and popular parks.

New geographical data or their deletion can be done at any time even after the text is published. In edit mode, a background mechanism keeps track of the reference text of existing geographical data when changes are made in the text and it finds their new position in the character strings in each paragraph. Therefore the links referring to the selected words or part of text for each geographical data are displayed correctly after the changes have been made. Obviously, if the text or parts of the text that refer to this data is erased, the relevant geographical data is lost forever.

3.6 Conclusions

The presented Web application presents a simple and universal to use means for authoring rich content through a Browser-based Web interface.

Besides content it allows for adding metadata. As such it will be the interface for adding any type of metadata including tasks as well.