Producing Location-Based Heritage Apps Using Only a ZIP File

GABRIEL WURZER, TU Wien, Austria
HANS RESCHREITER and FIONA POPPENWIMMER, Natural History Museum Vienna, Austria
JIRI UNGER, Czech Academy of Sciences, Czech Republic
CHRISTOPH LOBINGER and CHRISTIANE HEMKER, Archaeological Heritage Office of Saxony, Germany

Presentation of heritage may use a variety of media ranging from VR/AR to more classical ones such as video, images and text. During the course of the *VirtualArch Interreg Project*, the authors have implemented a framework that allows archaeologists and heritage experts to produce location-based cell-phone apps containing these kinds of media themselves, using only a ZIP file which is uploaded to a portal and automatically transformed into an app. However, eliminating the technical barrier is only one aspect that the project focuses on. More importantly, the presented approach reduces the time needed for app production to a matter of hours rather than months spent talking to app programmers who may or may not deliver a location-based app as intended by the archeologist or heritage expert who acts as a client; in our approach, there is no gap between client and app producer, both is done by the experts themselves, at no costs, using the content they choose to put on a map (e.g. images, videos, panoramas, 3D models and texts). Since this production process can be repeated for different target groups or even individual visitors, specialists can now offer a multitude of individualized location-based apps which can be constantly updated to reflect the state of the art. This paper presents the underlying concepts and technical details used by our approach, starting from conventions with ZIP file naming and contents, extending to map caching and custom tilesets being used, before discussing lessons learned during field tests.

Key words:
Heritage, Archaeology, Location-Based Apps, Transformation, Implementation.

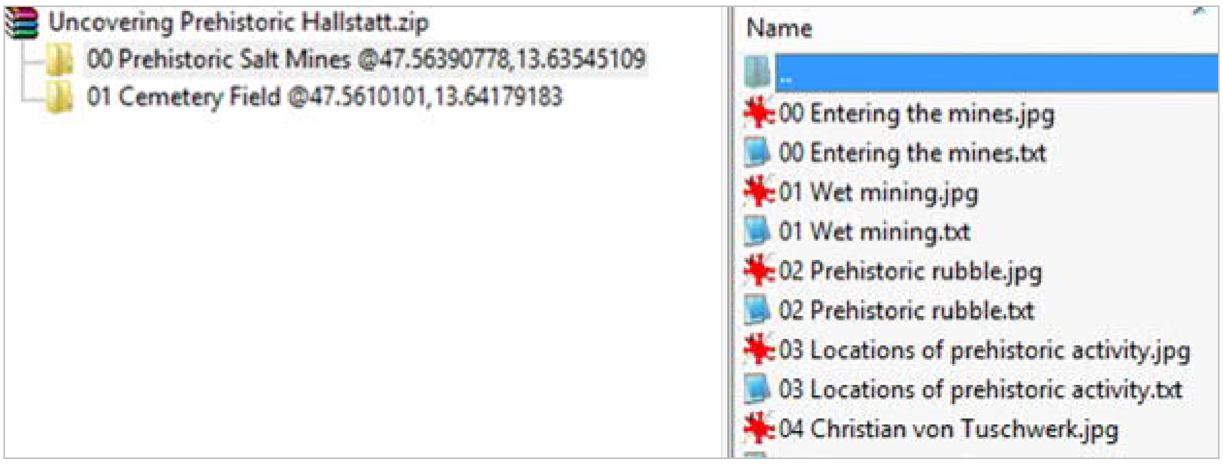
CHNT Reference:
Gabriel Wurzer et al. 2018. Producing Location-Based Heritage Apps Using Only a ZIP File.

INTRODUCTION

Producing cell phone apps requires ICT knowledge and money, both of which are limited in the fields of archaeology and heritage. Both problems have to be addressed in the *VirtualArch Interreg Project* [Lobinger and Hemker 2019], which required one location-based app for each of its eight pilot heritage sites. Training archaeologists with currently available app production tools was unfeasible, due to their highly technical nature and the fact that web mapping/visualization technologies evolve so rapidly that it is hard to keep up. Instead of teaching archaeologists and heritage experts how to program apps, the authors have instead sought a way to make them deliver *content* (i.e. media such as 3D files, videos, images/panoramas and text, packed as ZIP files; see Fig. 1) out of which a computer program can build a location-based app automatically. If this computer program is run as a free service in the cloud, then neither ICT knowledge nor money is required on the user's side. However, the complexities of that approach are not eliminated; they are merely delegated to a piece of processing infrastructure in the cloud. Numerous challenges – such as extracting and mapping the media items to points on a map, generation of a location-based app and compilation the a concrete cell phone operating system – need to be tackled, often using a

Authors addresses: Gabriel Wurzer, Research Unit of Digital Architekture and Planning, TU Wien, Karlsplatz 13, 1040 Vienna, Austria; email: gabriel.wurzer@tuwien.ac.at; Hans Reschreiter and Fiona Poppenwimmer, Department of Prehistory, Natural History Museum Vienna, Vienna, Austria; email: (hans.reschreiter, fiona.poppenwimmer)@nhm-wien.ac.at; Jiří Unger, Department of Prehistorical Archaeology, Czech Academy of Sciences, Letenská 4, 118 01 Praha 1, Prague, Czech Republic; email: unger@arup.cas.cz; Christoph Lobinger and Christiane Hemker, Archaeological Heritage Department Saxony, Zur Wetterwarte 7, 01109 Dresden, Germany; email: (christoph.lobinger, christiane.hemker)@lfa.sachsen.de

variety of processing techniques and programs that need to be orchestrated. To describe the details of a solution that deals with all these challenges thus forms the main contribution of this paper (see Sections 'ZIP to app' and 'Offline maps and custom tilesets'), which furthermore discusses lessons learned during field tests with a limited amount of users (see Section 'Discussion').


Uncovering Prehistoric Hallstatt.zip
00 Prehistoric Salt Mines @47.56390778,13.63545109
01 Cemetery Field @47.5610101,13.64179183

Name
..
00 Entering the mines.jpg
00 Entering the mines.txt
01 Wet mining.jpg
01 Wet mining.txt
02 Prehistoric rubble.jpg
02 Prehistoric rubble.txt
03 Locations of prehistoric activity.jpg
03 Locations of prehistoric activity.txt
04 Christian von Tuschwerk.jpg

Fig. 1. Sample contents of a ZIP file (© NHM Vienna)

ZIP TO APP

The steps used in the transformation of a ZIP file to an app were:

Production of a ZIP file which consists of a set of folders each standing for a "Point Of Interest" (POIs) on the map (refer to Fig. 1). These folders are numbered (e.g. "00"), named (e.g. "Prehistoric Salt Mines") and geo-tagged (e.g. "@47.56390778, 13.63545109"). They contain different media (e.g. images, videos, 3D models, panoramas) which are described in the following fashion: Each media item has a number (e.g. "00"), a name (e.g. "Entering the mines") and an extension. (e.g. .jpg, .png, .mp4, .3ds, .pano, .jpg). An (optional) textual description of this medium is to be found in an extra file that has the same number and name but the extension ".txt" (e.g. "00 Entering the mines.txt"). Within that description one may place plain text or HTML. The latter may be used for the inclusion of URLs, in order to link to further apps that show content in even richer detail or refer to additional information located online.

Upload of the ZIP file to the *VirtualArch Portal*: The authors have produced a web application called *VirtualArch Portal*, in which authorized users (e.g. user NHM acting on behalf of the Natural History Museum Vienna) may upload their ZIP file. The file is placed into a queue, due for processing in the background.

Server-side transformation of the ZIP into two web applications: The server-side transformation process starts by allocating a folder into which the portal copies two Javascript/CSS/HTML web applications which act as a template and are later customized for the specific content of the uploaded ZIP file: The first web application is for display of location-based media on a cell phone, the second for display of the same content in the web. Next step is the customization process of these two web applications on the server: A scanning process goes through each file and folder in the ZIP archive, in order to check whether it fits the numbering/naming nomenclature outlined before. If it does not, then this file or folder is considered a resource file (needed e.g. for textures when dealing with 3D models) and is copied immediately into the output folder. In all other cases, that media file is copied to the output using a generated name; its data added to a *JSON* file which contains all customization information – i.e. a list of POIs (physically stored as subfolders of the respective web application), file names of media items within that POI folders and, optionally, textual descriptions for each media item. This JSON file is then read by each of the two web applications on startup. The listed POIs are created as placemarks in a *Map View* and the media within these POIs are presented as a slideshow once the user clicks on a POI (*Media View*). A screenshot of both modes is given in the right upper part of Fig. 2. For being able to serve the map even when there is no internet connectivity, the transformation process also computes the bounding box of all POIs and downloads the associated map tiles into a subfolder which is used for offline viewing.

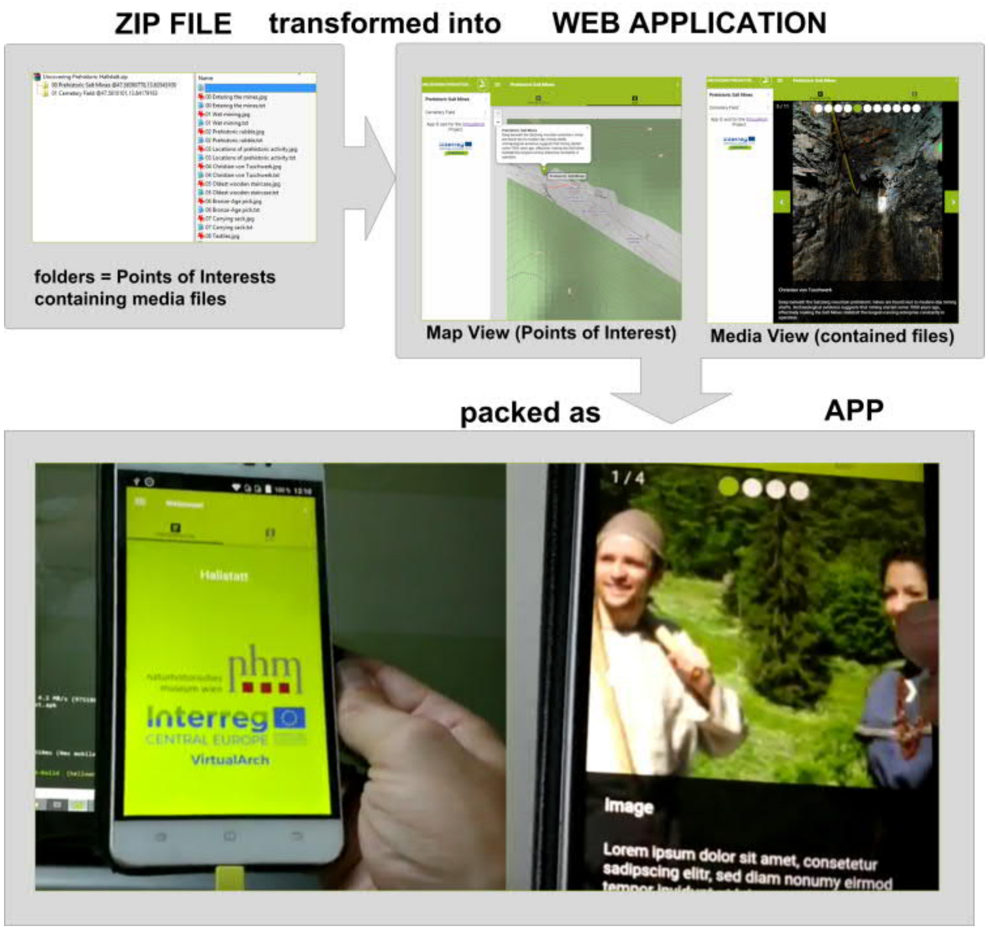

ZIP FILE
transformed into
WEB APPLICATION
folders = Points of Interests containing media files
Map View (Points of Interest)
Media View (contained files)
packed as
APP

Fig. 2. Overview of the transformation process from ZIP to location-based app (© NHM Vienna)

Packing as app: The web application for the cell phone must be styled so it looks like the target platform (i.e. *Android* or *iPhone*). This is done using the Framework *Onsen UI*[1], which can detect the platform it is running on and does all the necessary styling by itself. Using a cell phone's browser to view the web application would enable the user to see the POIs and Media, however, this (a.) requires internet connectivity which many tourists may not have and (b.) the browser itself diminishes the user experience, as users are used to click on an app which then opens in full screen mode rather than opening in a browser. Both problems are solved by a technology called *Apache Cordova*[2], which (1.) packs a folder containing a web application and media "into" an app (for *Android*; for *iOS*) so that it is available on the cell phone without internet connectivity (2.) uses a small web server also packed with the app to serve the web application from the same device, using (3.) a browser without address bar (i.e. *WebView*). After packaging, the app is stored in the portal.

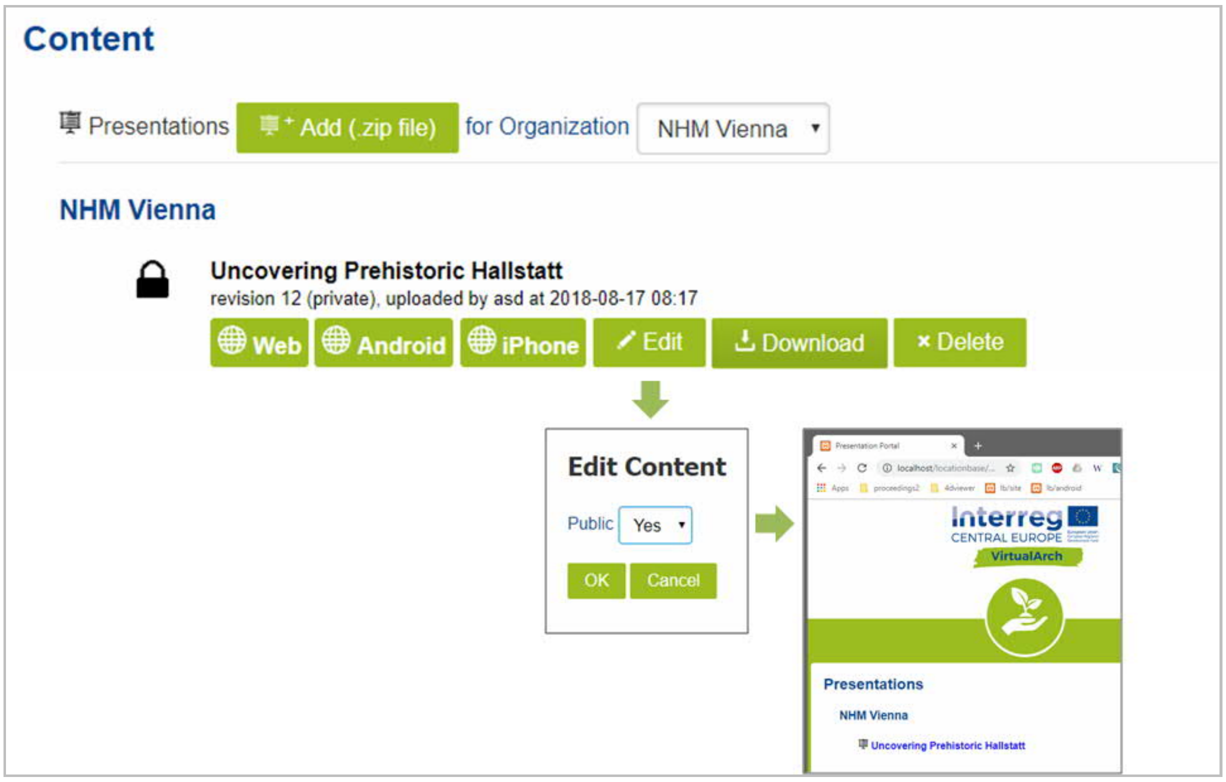
Transfer of the app to the phone. A user can download the app to a cellphone. Installing that app, however, requires that the user explicitly "trust apps from unknown sources" (currently only possible on *Android*). A far better alternative is to sign the app digitally and to afterwards upload it to *Google Play*/*App Store*, which

[1] https://onsen.io
[2] https://cordova.apache.org

(a.) enables users to download it without having to acknowledge whether he/she trusts this application and (b.) also guarantees that updates to the app will be delivered to the user. The downside of this approach is that it requires a hosting fee for the app, typically in the form of an annual fee (depending on the platform).

Opening the app on the cell phone is then just a matter of pressing the respective icon (see lower part of Fig 2). All pieces then fall into place: The tiny web server which *Apache Cordova* has packed into the app is started, a browser without address bar is pointed to the web application that the web server hosts, and the user sees a cell phone style user interface thanks to the automatic styling by *Onsen UI*.

Making the app available online. Given that the portal has also generated a web application for use on the internet, one may now use this in order to turn the portal into a “homepage”: Each transformed ZIP, can be set to “Public” (see middle in Fig. 3), which makes it appear on the front page of the portal. An (anonymous!) user may now access this “web presentation” (see lower right in Fig. 3), which means loading the respective web application for display on the web. All produced content can be furthermore downloaded from the portal: The initial zip file, the generated “web presentation” as well as the generated app(s). The portal itself will also become open source after it has stabilized, most probably distributed through a virtualization solution (e.g. *docker* or *VMWare* container).



Uncovering Prehistoric Hallstatt
revision 12 (private), uploaded by asd at 2018-08-17 08:17

Public: Yes

Fig. 3. Use of transformed ZIP as “web presentation” (© NHM Vienna)

OFFLINE MAPS AND CUSTOM TILESETS

An additional requirement set by the *VirtualArch Project* is that all content must be available offline, which includes the map tiles which are part of an app’s *Map View*. The reason for this is that internet connectivity might be lacking, either because the users do not have a data contract (e.g. tourists) or because a site is very remote. Map tiles are typically .png files stored on a server as folder hierarchy (*slippy map* standard, e.g. https://a.tile.openstreetmap.org/*z*/*x*/*y*.png, where *z* refers to a certain zoom level, *x* and *y* to the coordinates of a tile at that zoom level), which makes it possible to retrieve such files easily via http for offline storage.

The following process lets us download the map tiles for a set of POIs:

Computation of the bounding box: POIs are given by their latitude and longitude. The bounding box is computed by taking the minimum/maximum latitude and longitude. However, it could be that there is only one POI and thus the bounding box would have zero area. This situation is avoided by enlarging the bounding box, which has the additional advantage that it includes some of the surroundings. As methods for enlargement, two different modes were tested: (1.) Scaling by a factor (e.g. 10 %), which works well for reasonably small areas. However, it leads to heavy overhead in terms of tiles (and thus memory) when the area is large. A better method was to (2.) enlarge the bounding box by a “5-minute walk” (which is 416 m, given that a human has an average walking speed of 5 km/h [3.1 mph]). The inspiration for the second option came from municipal transport services (e.g. in Vienna), which use a “5-minutes walking radius” for depicting the surroundings of a station.

Download of the tiles: As a precondition, the zoom levels for download tiles must be known. The default value in this case are zoom levels 14 to 17, where tiles of zoom level 14 have a side length of 2.5km (adequate for rendering a whole archaeological site or village), while those on zoom level 17 have a side length of 300m (adequate for spots within that site or building ensembles within a village). For each zoom level, the bounding box is transformed from latitude/longitude coordinates into tile coordinates (tile x, tile y). Then the download happens using the previously mentioned URL schema http(s)://*server*/*z*/*x*/*y*.png. Most publicly available map servers are restricted with regards to how many requests they can serve per second. In the case of this work, the open mountain bike map[3] which is used by is often out of resources, and thus some tiles are not returned. The solution used by the authors is to (1.) retry failed requests at a later time and (2.) to cache downloaded tiles so they do not need to be downloaded in the first place. Likewise, there are restrictions on the requested zoom levels: Zoom level 17 and above (maximum is currently 19) are hard to get from public servers, since these suppose that such zoom levels are to be served from one’s own private server rather than straining public infrastructure. The authors have thus restricted access up to zoom level 17, but provide a way in which more zoom is possible (custom maps, presented in due course).

Serve tiles from local folder. The tiles within the bounding box are copied into each web application’s folder so that they can be served locally (i.e. offline, without download it from the internet). If a tile cannot be found because it failed to download, a blank tile will be shown. This situation could be alleviated by using a proxy script rather than pointing to the web application’s folder directly: The proxy first tries serving from the web application’s folder. If the tile in question could not be found, it tries downloading it from the internet instead. The authors did not implement this functionality however this would indeed be a path for future work. Also down-sampling from (existing) tiles on lesser zoom levels in place of (non-existing) tiles on higher zoom levels could be provided in such a proxy script.

Customized maps can be used to serve higher-resolution tilesets (i.e. zoom levels 18 or 19) or simply for mixing map content that is normally separate (e.g. sea and land maps), since this approach only supports one map layer. The process in that case would be as follows:

Assemble layers and export. In a GIS program (this project has used QGIS[4]), raster and vector layers are imported and arranged so that the overall map on screen is as intended for the purpose at hand. A map extract for the area in question is exported using a plug-in (e.g. QTiles[5]). The generated tileset must (a.) be in a folder [e.g. *my-custom-tileset*] and (b.) conform to the slippy map standard, i.e. the folder should have sub-folders with the *z*/*x*/*y*.png nomenclature.

Import into portal and use in a ZIP file. The exported tileset is zipped and uploaded into the portal. If an app should use this custom tileset, a special file *custom-map.txt* is to be placed into the generating zip file. The contents of that file give the name of the custom tile-set to be used e.g. my-custom-tileset]. The portal will look up that tileset and copy tiles from this location, as if they had been downloaded from the web and cached.

[3] http://mtbmap.cz/, served by Masaryk University, Czech Republic
[4] https://www.qgis.org
[5] https://plugins.qgis.org/plugins/qtiles

Some notes on extension points and caveats using these approaches:

> All available zoom levels of the custom tileset will be used. So if a custom tileset has zoom levels 10-19, these are the possible options for zooming in the Map View.
>
> The size of png files can be quite large compared to jpg files. Upon download/import of a custom tileset, files are automatically converted to .jpg (quality 50 %), in order to save space in the final app and thus on the user's cell phone. However, this comes at the expense of some graphical quality, since .jpg compression is lossy and compressing map tiles (i.e. line graphics) leads to lack of edge clarity, blurred textual labels and blocky artifacts within the images. A group of five test users were able to notice these encoding artifacts but did not explicitly mention them; to a different group of test users (n=6) two versions of the same app were shown, one using .png and one using .jpg files; all of them mentioned the difference in map quality as being clearly visible, however, when asked for the severity of that difference, they stated that the visual quality of the map was less the issue since they would use "Google Maps when they really wanted a map tool" instead.

DISCUSSION

Nowadays heritage presentation is using a lot of media for showcasing a site. However, there is even more material in the archives, which hardly gets presented at all. Using the methods presented in this paper, it is possible to resurrect these hidden treasures and to present them to a wider audience, without having to physically put them on display. A second point concerning such "hidden" heritage is that this can now be accessed in a location-based manner, e.g. when having a hike. The *VirtualArch Project* was targeting exactly this use case – showing people heritage that is buried beneath the earth (e.g. the prehistoric salt mines of Hallstatt/Upper Austria) or under water (e.g. the Roman harbour Barbir in Sukošan, City of Zadar). Some of these sites are furthermore too dangerous for a visit (e.g. the medieval silver mines of Dippoldiswalde, Saxony, which are in a state of collapse and thus need to be documented and filled with concrete), making it safer to explore the hidden/buried heritage above ground.

Having such a technical framework enables one to present archaeological content within a historical landscape; more input formats (e.g. 3D point clouds in addition to 3D models which can currently be imported; future file formats which are yet to be specified) and generated outputs (e.g. export to Augmented Reality glasses) can always be added if needed. But having so many options might also tempt archaeologists and heritage experts who are creating their apps themselves to add too many media types or too many media items. One rather has to reduce information and tell a story that visitors can relate to [Dobat et al. 2013], which has a lot more to do with museum education, storytelling and content preparation than technology.

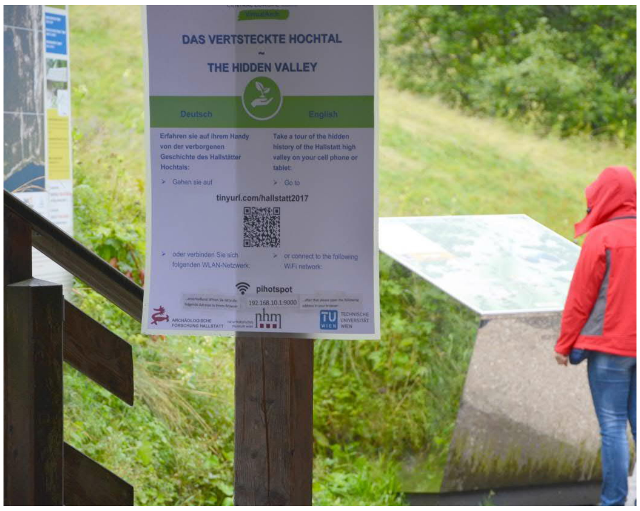

DAS VERTSTECKTE HOCHTAL
~
THE HIDDEN VALLEY
Deutsch
English
Erfahren sie auf ihrem Handy von der verborgenen Geschichte des Hallstätter Hochtals:
Gehen sie auf
Take a tour of the hidden history of the Hallstatt high valley on your cell phone or tablet:
Go to
tinyurl.com/hallstatt2017
oder verbinden Sie sich folgenden WLAN-Netzwerk:
or connect to the following WiFi network:
pihotspot
192.168.10.1:9000
ARCHÄOLOGISCHE FORSCHUNG HALLSTATT
nhm
TU WIEN
TECHNISCHE UNIVERSITÄT WIEN

Fig. 4. Delivering content via QRCode or Wi-Fi hotspot using captivate portal (photo from Hallstatt Science Fair 2017, © NHM Vienna)

On the technological side, there are also other options to choose from. When internet is available, presenting content using e.g. QR Codes situated directly at a POI is simple and works well (see Fig. 4). Integrating a wireless access point with the POI is another simple solution for giving tourists access to additional digital content. Another option is to turn the access point into a *captivate portal*: Upon entering the Wi-Fi network, the access point forwards the user to a predefined page which either located in the internet (if the access point is connected to it), or on a page on its local storage (when no internet connectivity is given). The latter case has been tested in the Hallstatt Science Fair (Upper Austria) by use of a Raspberry Pi which was powered by a cell phone power pack (2000 mAh). Even under adverse conditions (rain; fluctuating outdoor temperature between 5 and 20 degrees) the battery lasted two days. This approach was observed to work only when the user's browser wants to connect to a non-secure webpage once the contact with the WiFi network was established. The reason is that the access point has to fake the name resolution response (e.g. to http://www.nhm-wien.ac.at) so as to include its own IP address (e.g. 192.168.1.1) rather than the real one (*DNS masquerading*). If the request is secure and thus encrypted (i.e. https://www.nhm-wien.ac.at), then the access point cannot spoof a response. Recently almost all browsers – mobile or not – are trying to request via https rather than http.

CONCLUSION

This paper has presented a procedure to provide more information about heritage sites by transforming a ZIP file containing media into a location-based app. The intended audience ranges from heritage experts to tour guides who wish to show additional content. Since these groups are non-experts in programming, technical knowledge has been kept to a minimum – i.e. how to prepare a zip file and how to attribute files and folders within that. In this paper, an automatic transformation process that converts this ZIP into a location-based cell phone app was presented. Using this transformation, experts are now able to produce a multitude of location-based apps in a matter of hours, with practically no costs apart from hosting fees of the respective app stores. The presented workflow stands in direct contrast to now-common practice, where archaeologists and heritage experts must hire app programmers for producing their location-based apps. Apart from being costly, this entails communication problems and thus puts the quality of the produced result is at risk. By enabling archaeologist/heritage expert to produce location-based apps by themselves, the authors seek to remove this overhead and ensure that more location-based apps will be produced, and that these apps will be updated more frequently to reflect the current state of the art.

ACKNOWLEDGEMENTS

This work was made possible by the Interreg Central Europe programme, under the project "*VirtualArch – Visualize to Valorize*"[6].

REFERENCES

Christoph Lobinger and Christiane Hemker. 2019. The international project VirtualArch: visualization and presentation of hidden archaeological heritage across Central Europe. In *Proc. of the 23rd Conference on Cultural Heritage and New Technologies.* Vienna, Austria: Stadtarchäologie Wien, *to appear.*

Erik Dobat, Sandra Walkshofer, and Christoph Flügel. 2013. Mainlimes Mobil: Presenting Archaeology and Museums with the Help of Smartphones, in Nigel Mills (ed.). *Presenting the Romans: Interpreting the Frontiers of the Roman Empire World Heritage Site*, Suffolk: Boydell & Brewer and Boydell Press, Heritage Matters 12, 103–112

Imprint:
Proceedings of the 23rd International Conference on Cultural Heritage and New Technologies 2018.
CHNT 23, 2018 (Vienna 2019). http://www.chnt.at/proceedings-chnt-23/
ISBN 978-3-200-06576-5

Editor/Publisher: Museen der Stadt Wien – Stadtarchäologie
Editorial Team: Wolfgang Börner, Susanne Uhlirz
The editor's office is not responsible for the linguistic correctness of the manuscripts.
Authors are responsible for the contents and copyrights of the illustrations/photographs.

[6] https://www.interreg-central.eu/Content.Node/VirtualArch.html