Bildbetrachter - MyCoRe-Viewer

by Sebastian Hofmann, Kathleen Neumann
2016-01-14

Die in einer MyCoRe-Anwendung abgelegten Dokumente können z.T. auch wieder präsentiert werden entweder durch Anbindung einer Präsentationssoftware Dritter, oder durch den MyCoRe-eigenen Bildbetrachter.

Table of contents

1 Bildbetrachter - MyCoRe-Viewer

1.1 Beschreibung

Der MyCoRe-eigene Bildbetrachter ermöglicht es, unterstütze Dateitypen direkt in der Webseite zu präsentieren. Dies kann eingebettet, im Vollbildmodus oder auf mobilen Endgeräten genutzt werden. Unterstützt werden dabei verschiedene Bild-Formate (u.a. jpg, png, tiff) und PDF.

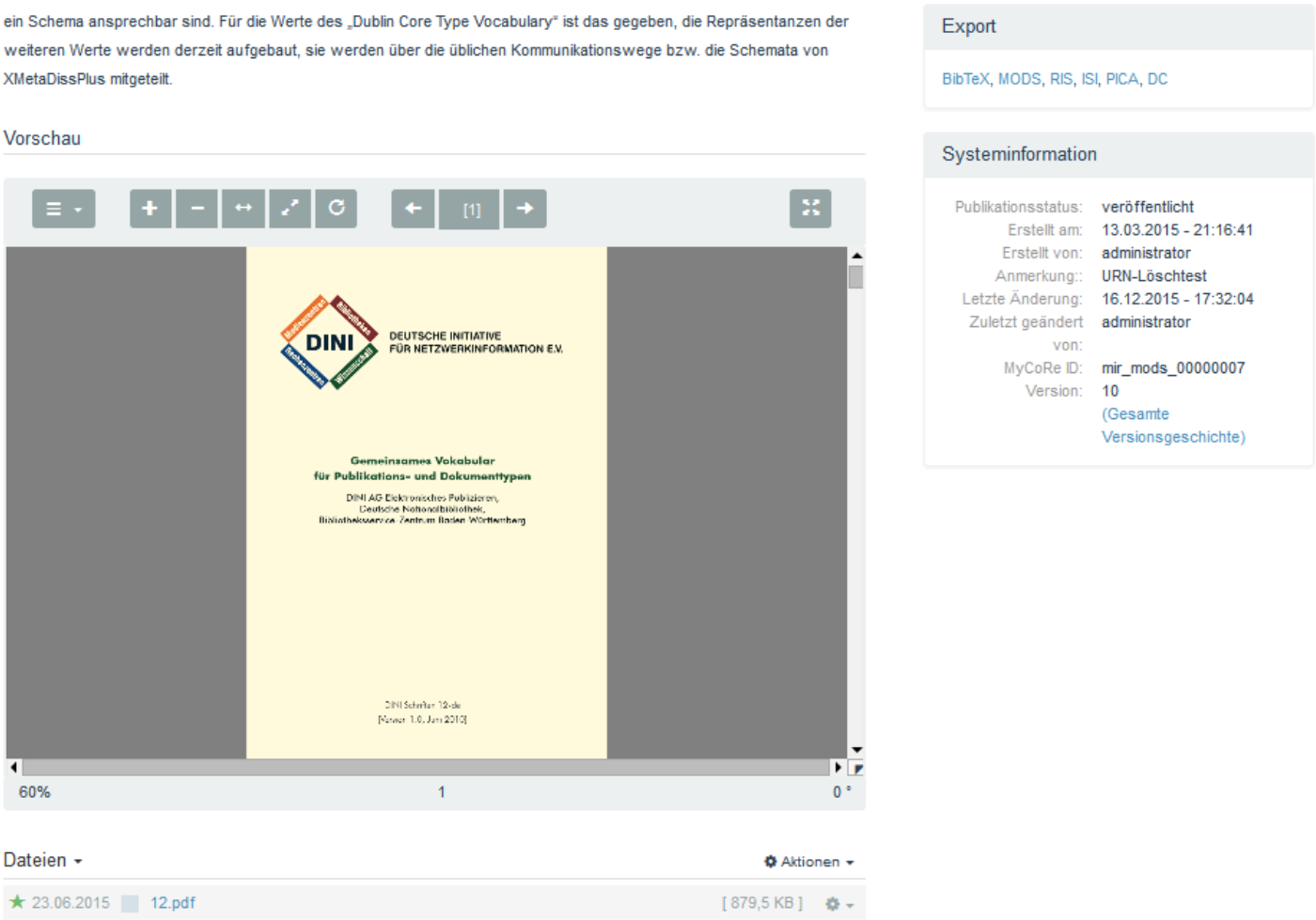

Abbildung: Screenshot einer MIR-Anwendung mit Bildbetrachter

1.2 Aufbau

Der MyCoRe-Viewer besteht aus verschiedenen Komponenten. Jede Komponente hat eine bestimmte Aufgabe. Diese Komponenten werden zu Modulen zusammen gebaut. Jedes Modul entspricht einer Javascript-Datei. Der MyCoRe-Viewer wird standardmäßig mit 9 Modulen ausgeliefert, jedoch lässt er sich einfach um weitere Module erweitern.

1.3 Wie binde ich den Viewer ein?

Zunächst müssen alle Javascript-Modul-Dateien und CSS-Dateien geladen werden (die URLs sind relativ zur BaseURL):

<script src="modules/iview2/js/iview-client-base.js"></script> <script src="modules/iview2/js/iview-client-desktop.js"></script> <script src="modules/iview2/js/iview-client-mets.js"></script> <link href="modules/iview2/css/default.css" type="text/css" rel="stylesheet">

Der MyCoRe-Viewer ist von JQuery abhängig:

<script src="//code.jquery.com/jquery-2.1.1.min.js"></script>

Der MyCoRe-Viewer in der Desktop Version ist abhängig von Bootstrap:

<script src="http://netdna.bootstrapcdn.com/bootstrap/3.3.5/js/bootstrap.min.js"></script> <link href="http://netdna.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css" rel="stylesheet">

Nun ist es möglich eine Instanz des Viewers zu erzeugen:

window["viewer"] = new mycore.viewer.MyCoReViewer(jQuery("body"), {});

Der erste Parameter des Konstruktors ist das Element in dem der Viewer erzeugt wird. Der zweite Parameter ist das Konfigurations-Objekt. Dieses Konfigurations-Objekt enthält die Konfiguration für alle Komponenten. Welche Komponente welche Konfigurations-Parameter benötigt wird im nächsten Abschnitt beschrieben.

Einfache Beispiele wie man den Viewer mit verschiedenen Konfigurationen einbindet findet man im MyCoRe-Viewer Projekt im Ordner src/main/resources/example/. Diese Konfigurationen lassen sich testen in den man im Basis-Ordner des MyCoRe-Viewer das Kommando mvn clean jetty:run ausführt. Anschließend wird der Viewer Kompiliert und ein Jetty-Server gestartet, welcher über den Port 9201 zu erreichen ist. Für das Kompilieren des Viewers ist es nötig dass Maven und Nodejs auf dem System installiert ist.

Für die Einbindung auf der Metadaten-Seite wird empfohlen das Frame-Modul anstelle des Desktop-Moduls zu benutzen.

1.4 Die neun Komponenten des Viewers

1.4.1 1. Das Base-Modul (obligatorisch)

Das Base-Modul ist der Kern des Viewers und enthält alle Komponenten die immer im Viewer benötigt werden. Alle anderen Module sind automatisch abhängig vom Base-Modul.

Das Base-Modul bietet folgende Konfigurationsparameter:

`mobile`	**(boolean)**

	gibt an ob die Instanz des Viewers eine Mobile-Version ist.
`doctype`	**(string)** gibt an welcher Dokumententyp geladen werden soll. (METS oder PDF)
`filePath`	**(string)** gibt an welche Datei geladen werden soll. Bei PDF der Link zur PDF Datei(relativ zum Derivat) und bei METS der link zum ersten Bild.
`derivate`	**(string)** das MyCoRe-Derivat, welches die Datei zum Anzeigen enthält.
`i18nURL`	**(string)** die URL zum I18N-Servlet. In der i18nURL kann {lang} als Platzhalter für das Sprachkürzel angegeben werden. z.B. "http://foo.bar/servlets/MCRLocaleServlet/{lang}/component.iview2.*"
`lang`	**(string)** gibt an welche Sprache der Viewer verwenden soll z.B. de
`webApplicationBaseURL`	**(string)** ist die URL zur MyCoRe-Instanz (damit der Viewer Ressourcen nachladen kann)
`onClose`	**(function)** gibt eine Funktion, an die beim Schließen des Viewers ausgeführt werden soll.

Folgende Parameter sind Optional:

`imageOverview.enabled`	**(boolean)** bestimmt ob die Übersichtsanzeige links neben dem Bild aktiviert ist
`chapter.enabled`	**(boolean)** bestimmt ob die Kapitelübersicht angestellt werden soll
`chapter.showOnStart`	**(boolean)** bestimmt ob die Kapitelübersicht bereits geöffnet werden soll, wenn der Viewer gestartet wird.
`permalink.enabled`	**(boolean)** bestimmt ob das Permalink Feature im Viewer aktiviert werden soll.
`permalink.updateHistory`	**(boolean)**

	wenn das Bild im Viewer gewechselt wird, dann wird auch in die URL im Browser geändert.
`permalink.viewerLocationPattern`	**(string)** ist das Pattern zum erstellen des Links zum Viewer default ist: {baseURL}rsc/viewer/{derivate}/{file}
`canvas.overview.enabled`	**(boolean)** bestimmt ob die Übersichtsanzeige rechts unten im Viewer angezeigt wird
`canvas.startup.fitWidth`	**(boolean)** legt fest, ob beim Starten des Viewers das Dokument in der Breite eingepasst wird (true) oder an die Viewergröße angepasst wird

1.4.2 2. Das Desktop-Modul

Das Desktop-Modul wird benötigt um den Viewer auf normalen Desktop-PCs darzustellen. Es orientiert sich an den CSS-Klassen und Javascript-Funktionen von Bootstrap und ist von diesem abhängig.

Keine Konfigurationsparameter.

1.4.3 3. Das Mobile-Modul

Das Mobile-Modul wird benötigt um den Viewer auf mobilen Geräten darzustellen. Es hat keine zusätzlichen Abhängigkeiten.

Keine Konfigurationsparameter.

1.4.4 4. Das Frame-Modul

Das Frame-Modul ist eine abgespeckte Variante des Desktop-Moduls. Es wurde speziell dafür entworfen den Viewer in eine extra Seite einzubinden.

Keine Konfigurationsparameter.

1.4.5 5. Das METS-Modul

Das Mets-Modul dient zur Anzeige von METS-Dateien und der zugehörigen Digitalisate über das IVIEW2 Interface. Es hat keine zusätzlichen Abhängigkeiten.

Das Mets-Modul bietet folgende Konfigurationsparameter:

`metsURL`	**(string)** die absolute URL zum METS-Dokument
`imageXmlPath`	**(string)** die absolute URL zum Iview-Interface
`tileProviderPath`	**(string)**

	eine Komma getrennte Liste von absoluten URLs zu den Iview-Interfaces

folgende Parameter sind Optional:

`pdfCreatorURI`	**(string)** die URL zum mets-printer (siehe mets-printer)
`pdfCreatorStyle`	**(string)** das Stylesheet welches für die Transformation des mets.xml benutzt werden soll
`pageRange`	**(number)** wie viele Seiten gedruckt werden können

1.4.6 6. Das PDF-Modul

Das PDF-Modul dient zur Anzeige von PDF-Dateien. Das Script PDF.js muss eingebunden werden, da der Viewer von diesem abhängig ist.

`pdfProviderURL`	**(string)** die URL zum PDF-Dokument {filePath} und {derivate} können als Platzhalter verwendet werden und werden vom Viewer durch die richtigen Werte ersetzt.
`pdfWorkerURL`	**(string)** PDF.js benötigt die pdf.worker.js. Diese wird automatisch nachgeladen und muss nicht extra in die Seite eingebunden werden.

1.4.7 7. Das Metadata-Modul

Das Metadata-Modul dient zur Anzeige von Metadaten überhalb der Strukturübersicht innerhalb des Viewers.

`metadataURL`	**(string)** die URL zu den fertig transformierten Metadaten(HTML). In der URL können {objId} und {derivateId} als Platzhalter eingesetzt werden.
`objId`	**(string)** die Objekt Id welche in die metadataURL eingesetzt wird

1.4.8 8. Das Logo-Modul

Das Logo-Modul ermöglicht die Darstellung eines Logos in der Toolbar.

`logoURL`	**(string)** die absolute URL zum Logo welches in der Toolbar angezeigt werden soll.

1.4.9 9. Das Piwik-Modul

Das Piwik-Modul bietet eine Anbindung an eine Piwik Instanz, dadurch kann verfolgt werden welcher Benutzer welches Digitalisat betrachtet hat.

`MCR.Piwik.baseurl`	**(string)** die URL zur Piwik Installation
`MCR.Piwik.id`	**(string)** die ID der Website in Piwik

1.5 Konfiguration der Viewer-Ressource

Der Viewer ist standardmäßig in jeder MyCoRe-Instanz eingebunden. Durch die URL %MyCoReBaseURL%/rsc/viewer/%derivateId%/%bild% kann er geöffnet werden. Diese Seite wird durch die MCRViewerResource gebaut. Die MCRViewerResource benutzt eine MCRViewerConfiguration um den Viewer zur verfügung zu stellen. Diese MCRViewerKonfiguration wird über das Interface MCRViewerConfigurationStategy zur verfügung gestellt. Welche MCRViewerConfigurationStrategy dafür benutzt wird kann über das Property MCR.Module-iview2.configuration.strategy gesteuert werden. Der Standard-Wert ist MCRViewerDefaultConfigurationStrategy.

Über die Klasse MCRViewerConfiguration kann gesteuert werden welche Script- und CSS-Dateien geladen werden, außerdem können Properties an die Viewer-Instanz gegeben werden. Für fast alle MyCoRe-Viewer-Module gibt es eine fertige MCRViewerConfiguration. Diese MCRViewerConfigurations erlauben es die Module über einfache MyCoRe-Properties zu steuern und übernehmen das einbinden der richtigen Script- und CSS-Dateien.

1.5.1 MCRViewerLogoConfiguration

`MCR.Module-iview2.logoURL`	wenn dieses Property gesetzt ist, dann wird das Logo-Modul automatisch eingebunden und konfiguriert.
`MCR.Module-iview2.logo.css`	dieses Property dient dazu zusätzliche Logos einzublenden.

1.5.2 MCRMetadataConfiguration

`MCR.Module-iview2.metadata.transformer`	bestimmt den Transformer der benutzt wird um das Objekt in HTML umzuwandeln. Es existiert ein vorgefertigtes Stylesheet für MODS-Metadaten.

1.5.3 MCRViewerPiwikConfiguration

Die Werte werden eins zu eins aus den MyCoRe-Properties durchgereicht (siehe Piwik-Modul).

1.5.4 Canvas-Konfiguration

`MCR.Module-iview2.canvas.overview.enabled`	disables the mini overview in the bottom right corner
`MCR.Module-iview2.canvas.overview.minVisibleSi`	this value specifies how large the viewer container must be (in px) to show the overview on startup. If the overview should not be shown by default the value can be set to a large value like 999999
`MCR.Module-iview2.canvas.startup.fitWidth`	if this property set to true the viewer will fit image width on startup

1.6 Kacheln als Basis

Wird der MyCoRe-Viewer für die Ansicht von Bildern genutzt, nutzt dieser die Bildkacheln, die standardmäßig beim Speichern der Daten im Hintergrund angelegt werden. Mehr dazu unter Grundlagen -> Derivate -> IView2 (../../documentation/basics/mcrderivate_ifs/iview2.html) .

1.7 Download

Der MyCoRe-Viewer ist eine eigene Bibliothek die einzeln heruntergeladen und unabhägig von MyCoRe genutzt werden kann. Die aktuelle Releasversion mit deren Neuerungen kann im Jira (https://mycore.atlassian.net/projects/MV?selectedItem=com.atlassian.jira.jira-projects-plugin:release-page&status=released#) eingesehen werden.

Zum Download stehen die jeweiligen Releases im MyCoRe-Artifactory (http://artifactory.mycore.de/webapp/#/artifacts/browse/tree/General/libs-releases-local/org/mycore/mycore-viewer) zur Verfügung.

2 DFG-Viewer

Am Beispiel des DFG-Viewer soll gezeigt werden, wie Bildbetrachter von Drittanbietern eingebunden werden können.