POM

Daniel Glazman

Publication Object Model #1

- Offer an abstraction layer hiding all the internal complexity of Publications
- Be Publication-format agnostic
- Three-layers architecture

Publication Object Model #2

- IPublicationManager
- IPublication
- IPublicationResource
- Extra interfaces for specific Publications

PublicationManager

- Register Publication formats
- List registered Publication formats
- Load a publication
- Create a publication

Publication

- All Publications implement a common interface known from the PublicationsManager
- But can implement public extras
- Edit and save Publications; manipulation of PublicationResources and metadata

PublicationResource

- Common interface for all resources inside a Publication
- Can be a Document, a binary file, whatever.

Thinking out loud…

```
let PM = PublicationManager;

PM.registerPublication("epub3", EPUB3Book);
// EPUB3Book implements IPublication

let ebook = PM.createPublication("epub3");

let author = ebook.setMetadata("author", "Daniel Glazman");
author.set("email", "daniel@glazman.org");

ebook.setMetadata("id",
                  "urn:uuid:0d98d561-f026-6844-b9e8-06e6b86e7c30");
ebook.setMetadata("scheme", "UUID");

let d = ebook.createResource("text/html", "main.html");
let cover = ebook.addResource("image/png", "cover.png");
cover.setMetadata("type", "cover");

ebook.save("file:///tmp/test"); // adds default file extension
                                // if not provided
```

TODO

- POM Spec
- EPUB3 as Publication spec
- OSS Multipurpose Framework, ideally in JavaScript and c++, possibly from common source

questions?