



Implementing a JSR-283 Content Repository in PHP

Karsten Dambekalns
karsten@typo3.org

ARRIVALS TO INTERNATIONAL PHP CONFERENCE

DATE	ARRIVING FROM	FLIGHT	GATE	DESTINATION
03 11	NEW YORK	IPC07	A12	FRANKFURT
03 11	BERLIN	IPC07	A34	FRANKFURT
03 11	BUKAREST	IPC07	B45	FRANKFURT
03 11	TORONTO	IPC07	C90	FRANKFURT
03 11	PARIS	IPC07	C23	FRANKFURT
03 11	ROMA	IPC07	A78	FRANKFURT
04 11	LONDON			

Introduction to JCR



Inspiring people to
share

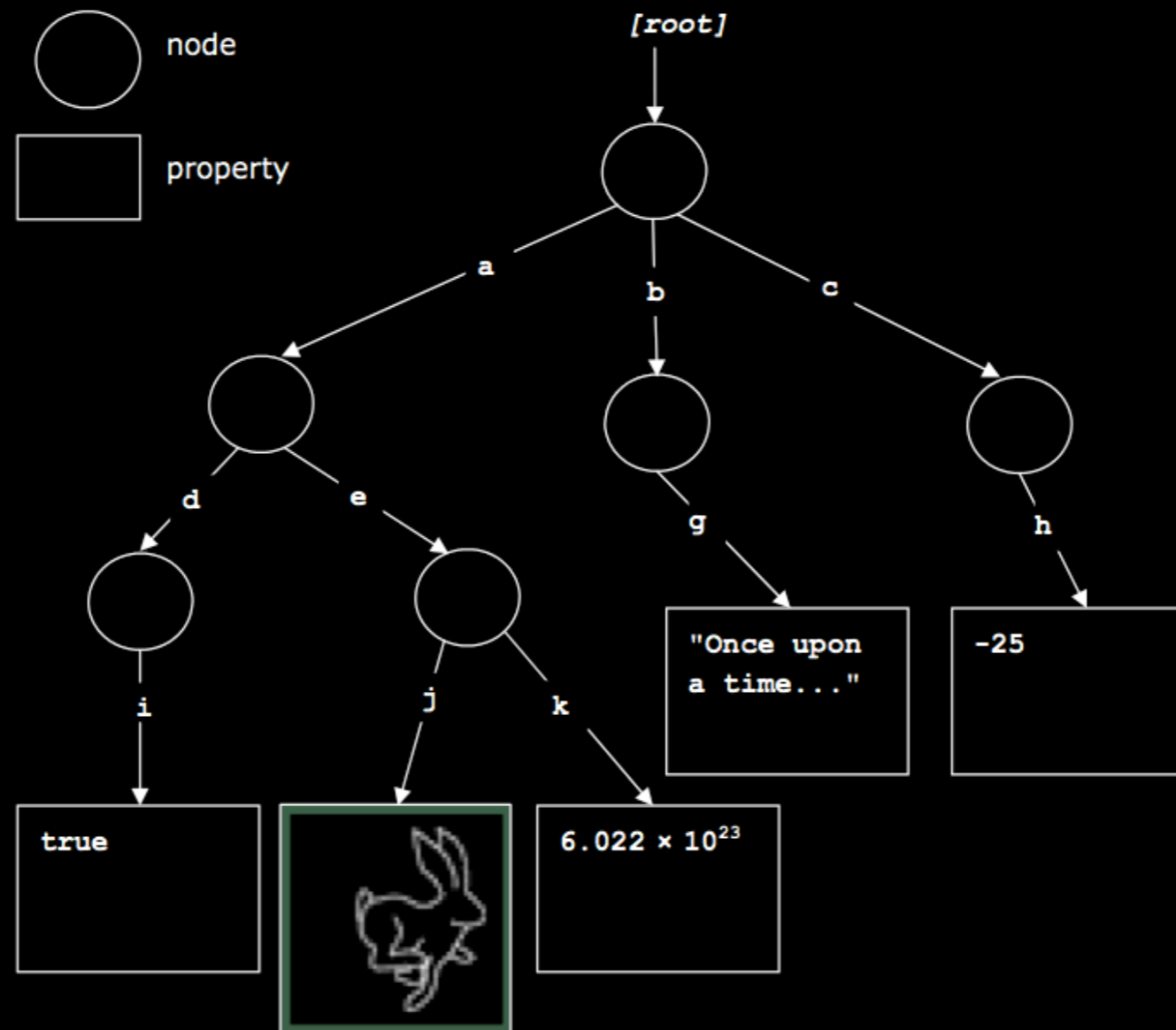
What is a Content Repository

- ❖ A Content Repository (CR) allows the storage and retrieval of arbitrary content as nodes and properties in a tree structure
- ❖ The node types as well as the tree structure can be freely defined by the user of the CR
- ❖ Binary content can be stored and queried as effectively as textual content
- ❖ The API for using a CR is standardized in the JSR-170 and JSR-283 specifications
- ❖ The API abstracts the actual data storage used (RDBMS, ODBMS, files, ...)



Inspiring people to
share

Nodes and Properties



Inspiring people to **share**

Reasons for using a CR

- ✔ Flexible and extensible data structure
- ✔ Object based storage and retrieval
- ✔ Combines advantages of navigational and relational databases
- ✔ Security can be enforced on a higher level
- ✔ Cleaner and easier to use for the developer
- ✔ No vendor lock-in
- ✔ Powerful data storage for a truckload of applications



Inspiring people to
share

Existing implementations

- ❖ Jackrabbit is the reference implementation, available as open source from the Apache Foundation
- ❖ Day CRX is the commercial CR implementation from the "inventor" of JSR-170, Day Software
- ❖ Other implementations are eXo JCR and Jeceira, the latter also being dead, and others
- ❖ JSR-170 connectors exist for Alfresco, BEA Portal Server, IBM Domino and others



Inspiring people to
share

PHP ports of the JSR-170 API

- ☛ Travis Swicegood ported the API to PHP in 2005 – project seems dead
- ☛ There is a port of the API available in the Jackrabbit sources, added 2005 – no relevant changes since then
 - This could be a sign of a well-done port of the API, but it rather seems to be a lack of usage and feedback
- ☛ No JSR-283 port of the API today
 - Well, since the specification isn't final yet, this may not come as a surprise...



Inspiring people to
share

TYPO3 Content Repository

- ❖ No standalone PHP implementation of a Content Repository exists as of now
- ❖ Our goal is a pure PHP implementation of JSR-283, but functionality needed for the TYPO3 CMS has priority
- ❖ Will take advantage of the TYPO3 Framework, but not be tied to the TYPO3 CMS.
- ❖ Could eventually become the standard CR for the PHP community?!
 - This depends also on you, the PHP developers out there...



Inspiring people to
share

Summary

- ☛ A Content Repository (CR) promises to solve a lot of the problems (not only) vendors of CMS currently have
- ☛ A stable standard with a fresh version in the making
- ☛ Various implementations exist, mostly in Java
- ☛ No standalone PHP implementation of a Content Repository exists

- ☛ We want to change that!



Inspiring people to
share

Development environment



Inspiring people to
share

TYP03 Framework

- ❖ Provides a robust and advanced programming framework with features like Dependency Injection, Aspect Oriented Programming, MVC, Component and Package Management, enhanced Reflection and more
- ❖ Inspired by the most popular frameworks and toolkits from Smalltalk, Python, Ruby and Java available today, picking the best concepts, skipping the annoyances
- ❖ Has already come a long way, check out the session by Robert Lemke today at 17:30 in Rhein-Main I!
- ❖ Not tied to TYP03 CMS, can be used for any PHP6-based project



Inspiring people to
share

PHP 6

- ❖ PHP 6 will be used for development of all projects around TYP03 5.0
- ❖ Currently unclear when PHP 6 will be released
- ❖ PHP 5.3 = PHP 6 - Unicode?
 - This could mean our code runs on PHP 5.3...
 - ... but will lack Unicode support
- ❖ Still needs to be investigated



Inspiring people to
share

Development model

- ☛ A few acronyms for your reading pleasure:
 - DDD – Domain Driven Design
 - TDD – Test Driven Development
 - CI – Continuous Integration
 - AOP – Aspect Oriented Programming
- ☛ We follow these principles and use those paradigms not because they sound good – but because they improve design and code
- ☛ Open Source, see <http://typo3.org/gimmefive/> for more



Inspiring people to
share

Summary

- ✔ The TYP03 Framework serves as a solid base for development
- ✔ PHP 6 will be a prerequisite for using the TYP03 CR
 - Following PHP roadmap changes we will evaluate possible escape routes
- ✔ Using state-of-the-art development principles



Inspiring people to
share

TYP03 Content Repository



Inspiring people to
share

Porting the JSR-283 API

- ❖ The API needs to be ported before writing the code
- ❖ Interfaces will not be ported up-front, but as we need them
- ❖ Facing typing issues, some Java types simply do not exist in PHP
- ❖ Binary data will probably be Resource Manager handles instead of streams
 - Alternatively simple stream handlers, still needs research
- ❖ Features prioritized according to the needs of the TYP03 CMS project



Inspiring people to
share

Actual data storage

- ☛ The underlying storage of the TYP03CR will be a RDBMS for the start
- ☛ Currently only SQLite through PDO is used
 - Easy to use for development and unit testing
 - The use of PDO already enables any PDO-supported database
- ☛ Specialized DB connectors will follow, using optimized queries, stored procedures, ...
 - This also opens the door for implementing other connectors, like to LDAP or object databases or ...



Inspiring people to
share

Data storage techniques

- ❖ Basically we need to store a simple tree
- ❖ Read access **must** be fast, as the majority of requests are read requests, write access **should** be fast
- ❖ Traditional approach as used – not only – in TYP03 today is to store a triplet (uid, pid, sorting)
- ❖ Alternative & faster method: Pre/Post Plane Encoding

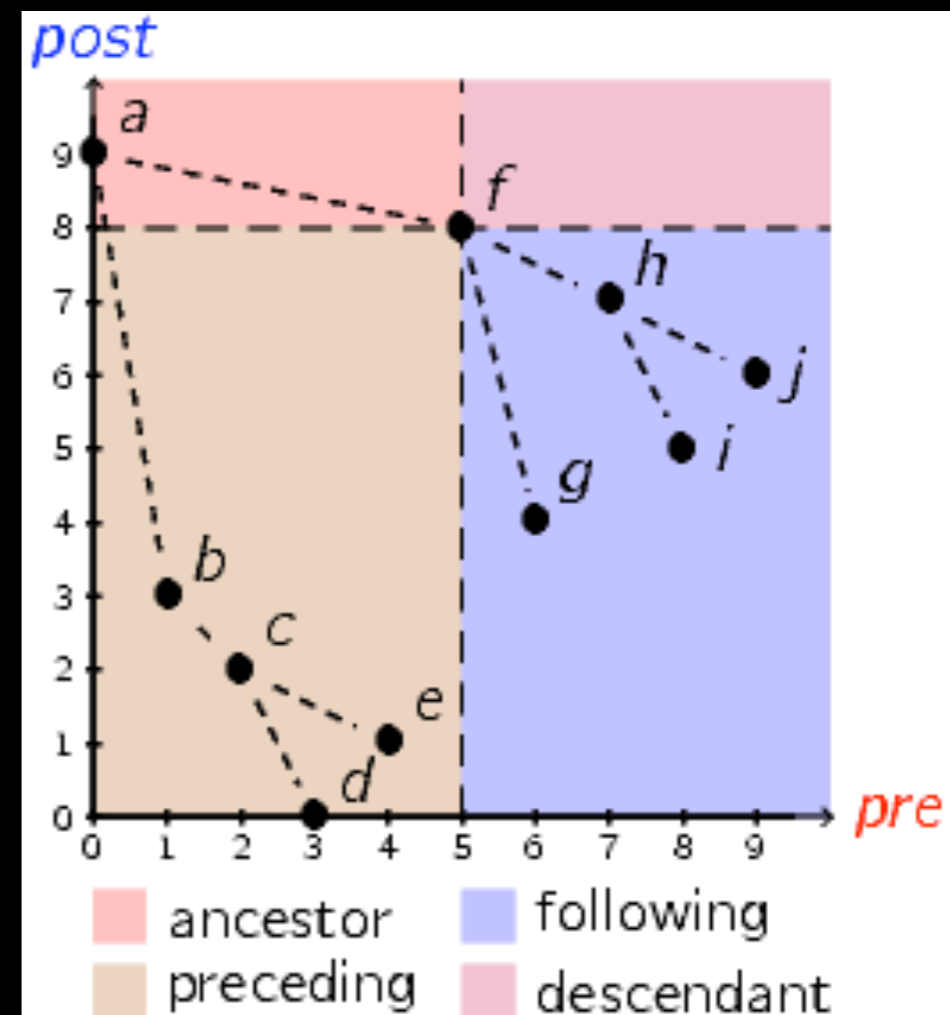


Inspiring people to
share

Pre/Post Plane Encoding

- Stores number determined by pre-order and post-order tree traversal
- Allows to partition the nodes into four regions, as shown for node f
- Very fast read access, e.g. a single SELECT to query all ancestors to a node f

SELECT * FROM table WHERE
 $pre < f.pre$ AND $post > f.post$



Inspiring people to
share

Performance considerations

- ♥ Nodes can already be fetched very fast
 - But attributes must be stored separately, to keep node table sizes inside sensible limits
 - Binary data will be stored in the underlying database for easier replication – a transparent cache to the filesystem will be added
- ♥ Write access can be sped up by various approaches like spacing and variable length indices for the pre/post numbers or by partitioning the data over more tables



Inspiring people to
share

Querying the TYP03 CR

- ♥ Using `getRootNode()` and friends from the API
- ♥ Using XPath queries
- ♥ Using SQL queries



Inspiring people to
share

API usage for querying

```
$repository = $this->componentManager->getComponent  
                ('T3_phpCR_RepositoryInterface');  
$session = $repository->login();  
  
$newsEntries = $session->getNodeByUUID  
                ('96bca35d-1ef5-4a47-8b0c-0bfc79507d08')->getNodes();  
  
foreach($newsEntries as $newsEntry) {  
    $title = $newsEntry->getProperty('title');  
    $text = $newsEntry->getProperty('text');  
    // more code here...  
}
```



Inspiring people to
share

XPath support for TYP03R

- ❖ To enable XPath we need
 - a XPath parser
 - an efficient way to transform a XPath query into SQL for the used low-level data structure
- ❖ The latter is a lot easier when storing the tree pre/post plane encoded



Inspiring people to
share

SQL support for TYP03R

- ☛ Using SQL we need
 - a (simple) SQL parser
 - an efficient way to transform that SQL into equivalent SQL for the used low-level data structure
- ☛ This still needs to be investigated, possible approaches include storing a reference to the parent node or even using the pre/post plane only as a cache for XPath read queries, optimizing the native storage for SQL read queries



Inspiring people to
share

Extensions to JSR-283

- ☛ A vendor may choose to offer additional features in his CR implementation
- ☛ The TYP03CR will offer support for
 - Data persistency through code annotations
 - Automatic node type generation based on class members
 - Rules for setting up virtual root nodes based on node types



Inspiring people to
share

Current status

- ❖ Currently the code supports a subset of the required features of levels 1 & 2 and the optional parts of the JSR-283 specification
 - Basic read & write access
 - Namespace registration
 - Node type discovery and registration
- ❖ Data storage uses the naive approach known from TYP03 4.x



Inspiring people to
share

Future plans

- ❖ Implementing missing required and optional features according to the JSR-283 specification
 - Workspaces and versioning will require special attention, as they are key features for the CMS as well!
- ❖ Implement native RDBMS connectors making use of specifically tuned queries and product-specific features
- ❖ Performance tests and tuning as needed



Inspiring people to
share

Summary

- ❖ Implementing the specification is not an easy task, but doable
- ❖ For the various parts a lot of research has already been done in the past
- ❖ The repository is a major improvement over the current way of storing data
- ❖ The whole PHP community could^Wcan^Wwill benefit!



Inspiring people to
share

Thanks for listening

Karsten Dambekalns <karsten@typo3.org>



Inspiring people to
share

TYPO3

