

TYPO3 services

What, why and how?

François Suter

Swiss TYPO3 Usergroup Conference, Olten, 24th January 2009

Services in TYPO3?

- introduced in 3.6
- used for authentication (FE, BE)
- used in the DAM
- part of the core
- a type of extension

Usefulness of services

- a generic way of implementing features that may have many variants
- allows differentiation for special cases
- available both FE and BE

Structure of a service

- Essentially a PHP class
- Required files in an extension:
 - *ext_localconf.php: service registration*
 - *sv1/class.tx_myext_sv1.php: PHP code for service*

Registering a service

- in `ext_localconf.php`
- using `t3lib_extMgm::addService()`

Parameters	Notes
<code>\$extKey</code>	The extension key
<code>\$serviceType</code>	The type of service
<code>\$serviceKey</code>	Unique key for the service
<code>\$info</code>	Detailed information about the service

Service details

Parameters	Notes
title	Name of the service
description	Detailed description of the service
subtype	Subtypes that the service can handle
available	Whether the service is available or not (true or false)
priority	The priority of the service
quality	The quality of the service
os	The OS required for the service to work properly
exec	Any executables the service relies on
classFile	The name of the file the service is found in
className	The name of the class to call for the service

Refining a service configuration

- Override an existing configuration
 - *In `typo3conf/localconf.php`*
 - *Syntax*
 - > `$TYPO3_CONF_VARS['T3_SERVICES'][type][key][property] = value;`
 - *Example:*
 - > `$TYPO3_CONF_VARS['T3_SERVICES']['auth']['tx_sv_auth']['priority'] = 90;`
- Adding configuration options
 - *In `typo3conf/localconf.php` or any `ext_localconf.php`*
 - *Syntax:*
 - > `$TYPO3_CONF_VARS['SVCONF'][type][key][property] = value;`
 - > `$TYPO3_CONF_VARS['SVCONF'][type]['default'][property] = value;`
 - *Retrieved using `t3lib_svbase::getServiceOption()`*

Refining a service configuration (2)

- Options outside the service
 - *Not used inside the service, but by code calling the service*
 - *Syntax:*

```
> $TYPO3_CONF_VARS['SVCONF'][type]['setup'][property] = value;
```
 - *Example:*

```
> $TYPO3_CONF_VARS['SVCONF']['auth']['setup']['FE_alwaysFetchUser'] = value;
```
 - *No API for this, usage as per documentation*

Service-related API

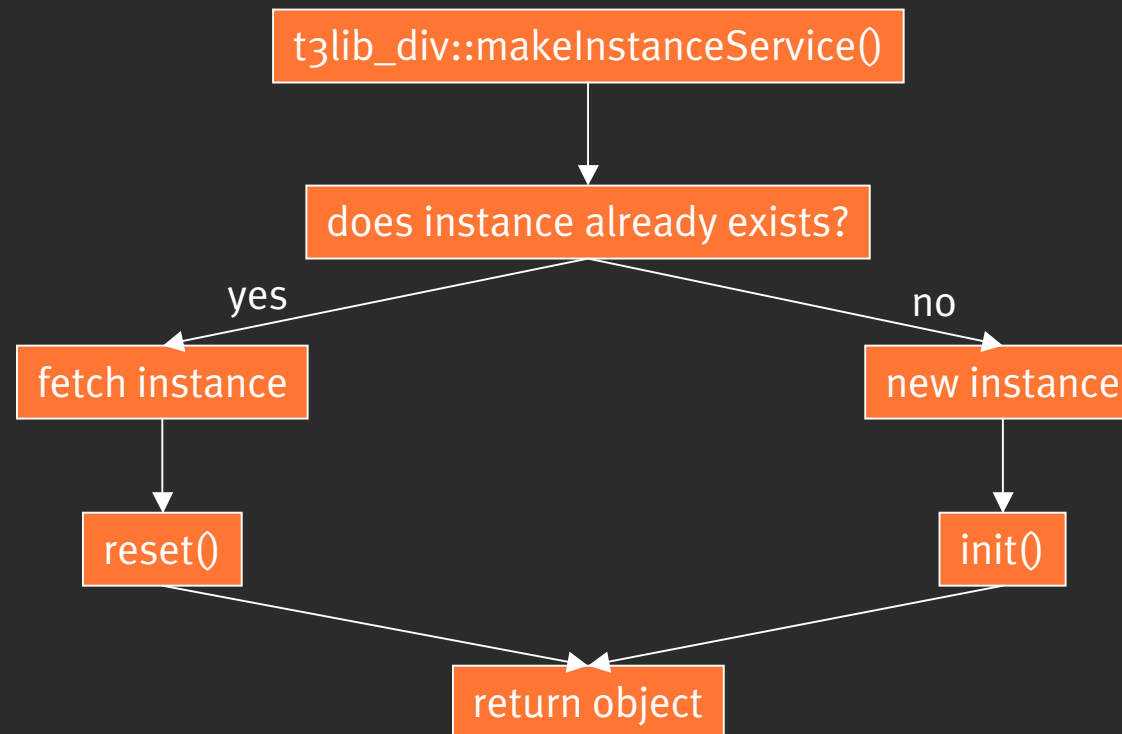
- In `t3lib_extMgm`:

Method	Usage
<code>addService</code>	Used to add a service to the list of services. Stores all in the info in <code>\$T3_SERVICES</code> .
<code>findService</code>	Finds a specific service for a given type and subtype, taking into account priority, quality and availability.
<code>deactivateService</code>	Temporarily marks a service as unavailable.

- In `t3lib_div`:

Method	Usage
<code>makeInstanceService</code>	Finds an appropriate service given a type and a subtype.

Getting a service instance



Using a service chain

- Call all services of a given type/subtype

```
$exclude = '';  
while (is_object($serviceObj = t3lib_div::makeInstanceService($type, $subtype, $exclude))) {{  
    $exclude.= ','.$serviceObj->getServiceKey();  
    // Do something  
}}
```

- Example: auth services
 - *authUser()* called inside a service chain
 - *continuation of chain depends on return code*

Service API

- Base class `t3lib_svbase`
 - *Getter methods for service information*
 - *Error handling*
 - *I/O tools for reading and writing files*
 - *Service implementation*
 - > `init()`
 - > `reset()`
 - > `__destruct()`

Implementing a service

- Existing type
 - *Inherit the base class (if any)*
 - *Implement the relevant API*
- New type
 - *Create a base class*
 - *Define own API in base class*

Looking forward

- Coming soon to a TER near you:
 - *Revamped core documentation about services (doc_core_services)*
- Clean up of service-related API
 - *Store instances in static variable instead of global*
- BE information module (maybe)
 - *Inspired by the DAM's info module*