# Contenido Extension Chainer

Project: Contenido

Updated: 2004-10-21

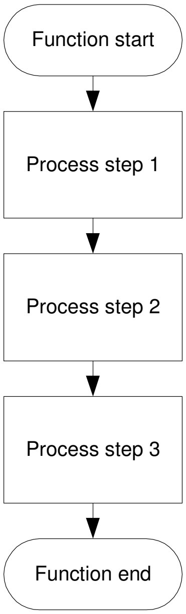
# 1. Short description

The goal is to create a facility to allow developers to extend Contenido base functionalities or Contenido Plugins for projects.

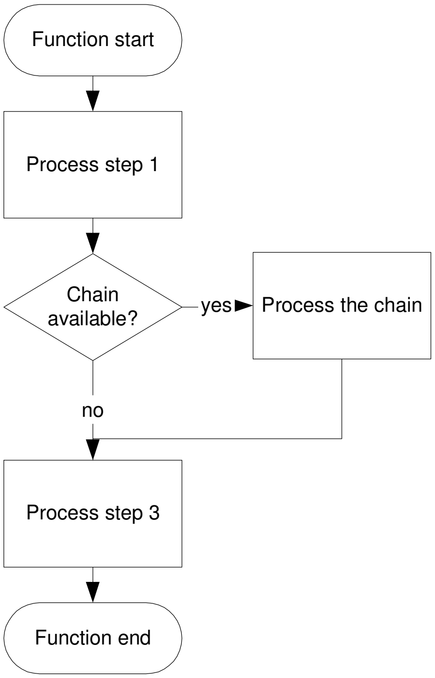
The solution is the development of so-called "Contenido Extension Chains" (short: "CEC"). A CEC provides a facility to extend a process using a so-called "Chain Function".

A chain is a an extension facility to extend or modify an existing process without chaning the base functionality with "hacks".

Old behaviour in processes



New behaviour in processes (Chain)



## 2. Change details

The change includes a few new classes to implement the "CEC" functionality. The base class is the "Chain Registry" which manages all chains.

The chain registry is created within the global scope of Contenido, so it is available virtually everywhere. The following terms are introduced:

| | |
|---|---|
| Extension Chain: | A chain of functions (none, one, more) which are responsible for processing the data. |
| Chain Name: | Unique name of the chain |
| Chain Registry: | Class to manage all chains |
| Chain Function: | One single function to process data |

### 2.1. CREATION OF THE CHAIN REGISTRY TROUGH CONTENIDO

Contenido handles the creation of the chain for front-and backend equally:

```
$_cecRegistry = new cApiCECRegistry;
```

The chain registry will be accessible via $_cecRegistry.

### 2.2. NAMING CONVENTIONS

Each chain has to have an unique name. To reduce the risk of duplicates, the following naming conventions are set:

- The name of a chain has to be <type>.<area>.<functionality>
- "Type" is either "Contenido" for Contenido base functions or the project's name for project specific chains.
- "Area" is the area in which the chain is used, e.g. "Content" in Contenido.
- "Functionality" is a clear text name to describe which functionality is changed (or modified)

## 2.3. CREATING A CHAIN FACILITY WITHIN CONTENIDIO (CONTENIDO DEVELOPERS)

The method "registerChain" has to be called to define the chain name and the parameters:

```
$_cecRegistry->registerChain("Contenido.Content.SaveContentEntry", "int", "int", "int",
"string");
```

The first parameter of "registerChain" contains the name of the chain and is mandatory. All other parameters define the type of the parameters.

The function "registerChain" registers a new chain in the system and defines which parameters the chain functions are receiving.

All calls to "registerChain" in the Contenido base have to be inserted into the configuration file "config.chains.php". All calls to "registerChain" have to be inserted into the file "config.chains.plugin.php".

**Important: Once a chain was registered with Contenido, the number, the order and the types of a chain are not allowed to change!**

## 2.4. EXECUTION OF A CHAIN (CONTENIDO- OR PLUGIN DEVELOPERS)

A chain can be executed via retrieving the chain iterator, and then calling "execute" on each chain entry.

Example:

```
$_cecIterator = $_cecRegistry->getIterator("Contenido.Content.SaveContentEntry");

while ($chainEntry = $_cecIterator->next())
{
    $value = $chainEntry->execute($idartlang, $type, $typeid, $value);
}
```

**Important: The developer who implements a chaining facility has to document what a chain function is allowed to do. In our example, it would be okay to manipulate the content, but a charset-conversion would be invalid.**

## 2.5. INSERTING A CHAIN FUNCTION INTO THE CHAIN (EXTENSION DEVELOPER)

The developer who wants to extend a chain has to call "addChainFunction" with the chain name and the chain function name..

Example:

```
$_cecRegistry->addChainFunction("Contenido.Content.SaveContentEntry",
"cecSaveContentEntry_EncryptEmailAsForm");
```

**Important: The chain function has to work, no matter in which position the chain is placed.**

**Important: Chain functions may not interfere with other chain functions in the same chain.**

# 3. Contenido Extension Chains for Decision Makers

## 3.1. WHEN DOES IT MAKE SENSE TO USE CHAINS FOR PROJECTS?

The use of CEC's makes sense when it is not known yet if functionality has to be inserted at a specific code location or not or if it should be possible to extend existing functionality later. It only has to be defined at which location functionality should be extended.

## 3.2. WHEN DOES IT MAKE SENSE TO USE CHAINS FOR CONTENIDO?

The use of CEC's within Contenido makes sense if the base functionality of Contenido could be changed in a project without rewriting code of Contenido itself. If the CEC is placed at the correct location, each project can access these chains.

## 3.3. WHAT ARE THE RESTRICTIONS FOR CHAINS?

CEC's should only be used for data manipulation. They should not be used to manipulate GUI elements, as the GUI stuff within Contenido will be changed soon.

# 4. Contenido Extension Chains for Developers (Contenido / Customer Projects)

## 4.4. WHEN DOES IT MAKE SENSE TO USE CHAINS WITHIN PROJECTS?

The implementation of a CEC within a customer project makes sense if the Project is "generic" and/or can be used in different projects, but with different detail functionalities.

## 4.5. WHEN SHOULD A CHAIN BE IMPLEMENTED WITHIN CONTENIDO?

Please contact a Contenido developer if you like to have a chain implemented within Contenido, or implement the chain and send the patch to the Contenido developers.