Joomla! 2.5

Capire i Template

Guida estremamente basilare per iniziare a realizare template

[Una guida molto semplice]

Ciao a tutti,
iniziamo oggi un percorso che ci porterà a comprendere meglio il modo in cui creare un template basilare ma “accettabile” per Joomla! 2.5
Non realizzeremo un template distribuibile in quanto non potremo prendere in considerazione tutto. Questa guida vi aiuterà solamente a prendere confidenza con il modo in cui si crea un template per Joomla e quali sono le possibilità che questo software ci offre.
Non ci occuperemo quindi di funzionalità troppo avanzate, per le quali esistono numerosi tutorial (anche se purtroppo in italiano la scelta è molto ridotta), ma ciò che tenterò di fare è dare una base di aiuto a chi si affaccia adesso a questo splendido universo di nome Joomla!

[Gettin Started]

Prima di tutto è necessario dire che per sviluppare un buon tema è necessario testarlo continuamente su un’installazione di joomla! correttamente funzionante, ergo: se non ne avete una vedete di provvedere!
Un ulteriore consiglio che mi sento di dare a chi si avvicina a Joomla! è quello di spendere del tempo nella conoscenza del software stesso dal punto di vista dell’utente. La prima volta che si installa una versione di Joomla è importante installare anche i contenuti di esempio, che saranno un validissimo aiuto all’apprendimento delle funzionalità e della logica che sta alla base di questo potentissimo software. Realizzare un buon template è praticamente impossibile senza questo propedeutico tirocinio. Fidatevi, non sarà stato tempo sprecato!
Tornando a noi, una volta reperito ed istallato joomla! (nella versione 2.5) è il momento di dargli un’occhiata dal di dentro. Attraverso un client FTP, specie se si lavora in remoto, sarà possibile navigare all’interno della directory root di Joomla!
Come avrete modo di vedere, la directory principale contiene al suo interno altre directory ed alcuni file. Per capire a cosa servano vi rimando alla ricca documentazione reperibile in rete.

Prima di iniziare a sviluppare un template, prenditi tempo per conoscere meglio Joomla!
Non sarà fatica sprecata!

Ciò che a noi interessa in questa sede è solamente la directory “templates” la quale contiene al suo interno i vari template installati, anch’essi contenuti a loro volta in ulteriori sottodirectory.
Il template che creeremo andrà a finire proprio in questa directory … esattamente come tutti gli altri.
Ma abbandoniamo adesso il core di Joomla! Per ritornare a concentrarci sullo sviluppo vero e proprio del template.

[Un nome un Template]

Per iniziare scegliamo un nome da dargli che sarà anche il nome della prima directory che si dovrà creare. Io ho scelto il nome DonkTemplate.
Creiamo quindi sul nostro pc una directory dal nome “donktemplate”.
Dopo questo primo step, sarà necessario creare alcune altre directory e file. **Le directory da creare necessariamente sono**:

1. **css**
2. **images**

I file da creare nella directory principale sono (lasciandoli momentaneamente bianchi):

1. **index.html**
2. **index.php**
3. **templateDetails.xml**

Come ultimo passo, si dovrà creare un file dal nome template.css e salvarlo all’interno della directory “**css**”.

Fatto questo sarà possibile dare il via alla fase di sviluppo. Come sarà facile intuire i file sui quali dobbiamo agire all’inizio sono essenzialmente *index.php* e *css/template.css* attraverso i quali disegneremo un layout grafico per il sito esattamente come se stessimo sviluppando una comune pagina html.
Il file *index.php* conterrà il codice (x)html mentre le definizioni di stile css saranno inserite nel file *css/template.css* .

Albero delle directory del Template in costruzione

donktemplate

images

css

Per sviluppare un buon template puoi utilizzare anche soltanto un semplice editor di testo!

Ciò che a noi interessa in questa sede è solamente la directory “templates” la quale contiene al suo interno i vari template installati, anch’essi contenuti a loro volta in ulteriori sottodirectory.
Il template che creeremo andrà a finire proprio in questa directory … esattamente come tutti gli altri.
Ma abbandoniamo adesso il core di Joomla! Per ritornare a concentrarci sullo sviluppo vero e proprio del template.
Nel caso di questo esempio, la struttura della pagina sarà molto elementare e priva di accorgimenti estetici in quanto più semplice da gestire per un tutorial.

Ecco quindi un po’ di codice da utilizzare.
Non entrerò nel merito del codice in quanto non è materia di questa guida.

Contenuto del file template.css

```
body {
     margin: 0px;
     padding: 0px;
     font-family: Arial, Helvetica, sans-serif;
     font-size: 14px;
     color: #FFFFFF;
}
a {}
a:hover {}
h1 {}
h2 {}
h3 {}
#all_joomla {
     background-color: #000000;
     width: 1000px;
     margin-right: auto;
     margin-left: auto;
     overflow: auto;
}

#header {
     float: left;
     width: 960px;
     padding-right: 20px;
     padding-left: 20px;
}

     #logo {
     float: left;
     width: 225px;
     height: 60px;
}
#nomesito {
     float: left;
     width: 725px;
     padding-left: 10px;
     height: 60px;
}
```

```
     #nomesito h1 {
     margin: 0px;
     padding: 0px;
}
     #search_container {
     float: left;
     width: 215px;
     padding-right: 10px;
}
     #pathway_container {
     float: left;
     width: 725px;
     padding-right: 10px;
     font-size: 10px;
}

#page {
     float: left;
     width: 960px;
     padding-right: 20px;
     padding-left: 20px;
     padding-top: 20px;
     margin-bottom: 20px;
}
#colonna_sinistra {
     width: 178px;
     float: left;
     padding-top: 10px;
     padding-right: 10px;
     padding-bottom: 20px;
     padding-left: 10px;
     border: 1px solid #999999;
}
#colonna_centrale {
     float: left;
     width: 498px;
     margin-right: 20px;
     margin-left: 20px;
     border: 1px solid #999999;
     padding: 10px;
}
#colonna_destra {
     width: 178px;
     float: left;
     padding-top: 10px;
     padding-right: 10px;
     padding-bottom: 20px;
     padding-left: 10px;
     border: 1px solid #999999;
}
#footer{
     float: left;
     width: 938px;
     border: 1px solid #999999;
     margin-right: 20px;
     margin-left: 20px;
     padding: 10px;
     margin-bottom: 50px;
}
```

Ricorda che questo codice è solo una bozza di template, non ancora funzionante in Joomla!
... e poco anche come semplice pagina html

Contenuto del file index.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
<title>DonkTemplate</title>
<link href="css/template.css" rel="stylesheet" type="text/css" />
</head>

<body>
<!-- CONTENITORE GENERALE -->
<div id="all_joomla">

  <!-- HEADER -->
  <div id="header">

    <div id="logo"><img src="images/logo.png" alt="Logo" /></div>

    <div id="nomesito">
      <h1>Nome Sito</h1>
      descrizione del sito
    </div>

    <div id="pathway_container">Patway Breadcrumb</div>

    <div id="search_container">Search Box</div>

  </div>
  <!-- /HEADER -->

<!-- COLONNE -->
  <div id="page">

     <div id="colonna_sinistra">Colonna Sinistra</div>

    <div id="colonna_centrale">Colonna Centrale</div>

    <div id="colonna_destra">Colonna Destra</div>

  </div>
<!-- /COLONNE -->

<!-- FOOTER -->
     <div id="footer">Footer</div>
<!-- / FOOTER -->

</div>
<!-- CONTENITORE GENERALE -->
</body>
</html>
```

[Attenti alla testa!]

Come ho già detto non è questa la sede per discutere di html e css e quindi non prenderò in esame la pagina se non come uno spazio grafico suddiviso in parti.
Ho suddiviso questo spazio in modo da poter ospitare alcuni contenuti che saranno generati da Joomla!
La domanda infatti è questa: come facciamo a trasformare questa semplice pagina html in un vero e proprio template che possa interagire al meglio con Joomla?
Per iniziare questa trasformazione dovremo riscrivere nuovamente il file *index.php* iniziando dall' head della pagina.
Le seguenti righe di codice ci permettono di iniziare un'analisi su come joomla! Si "rapporta" con ogni suo template.

```
<?php
// No direct access.
defined('_JEXEC') or die;
/* The following line gets the application object for things like displaying the site name */
$app = Jfactory::getApplication();
?>
```

Queste prime righe in php aprono il codice della pagina e sono una delle innovazioni introdotte nel passaggio dalla versione 1.5 alle nuove 1.6 e successive. Il loro scopo è quello di aumentare la sicurezza del sito. Non ci soffermeremo su questo, in quanto si tratta di un tema troppo avanzato per questa guida. La cosa importante è capire che esse sono necessarie e devono necessariamente essere inserite in testa al file index.php

Continuando con l'esplorazione del codice incontriamo altre righe

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="<?php echo $this->language; ?>" lang="<?php echo $this->language; ?>" >
<head>
<jdoc:include type="head" />
<link rel="stylesheet" href="<?php echo $this->baseurl ?>/templates/system/css/system.css" type="text/css" />
<link rel="stylesheet" href="<?php echo $this->baseurl ?>/templates/system/css/general.css" type="text/css" />
<link rel="stylesheet" href="<?php echo $this->baseurl ?>/templates/<?php echo $this->template ?>/css/template.css" type="text/css" />
</head>
```

Questa parte di codice è sicuramente più intuibile in quanto presenta alcuni elementi propri di una pagina xhtml. La prima riga infatti non è altro che una comune apertura nella quale si dichiara il linguaggio della pagina.
La riga:

```
<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="<?php echo $this->language; ?>" lang="<?php
echo $this->language; ?>" >
```

è invece l'impostazione della lingua per la codifica che ne dovrà fare il browser. Da notare che al posto del nome della lingua viene inserita una variabile richiamata attraverso un script in php **<?php echo $this->language; ?>**. Questo avviene perché Joomla è stato da sempre pensato come un prodotto adattabile anche dal punto di vista delle lingue.
Proseguendo ancora incontriamo altre righe. Ci troviamo ora nell'head vero e proprio della pagina. Le righe in questione si aprono infatti con il tag <head> e si chiudono ovviamente con il tag </head>. Ma analizziamo ora ciò che è contenuto tra i due tag.

```
<jdoc:include type="head" />
```

Questa riga, anch'essa parte del funzionamento di Joomla, va a sostituire quelli che comunemente chiamiamo metatag, solitamente utilizzati per fornire informazioni sulle pagine agli utenti o ai motori di ricerca.
Questo modo ci permette di gestire questi tag direttamente dal pannello di amministrazione di Joomla dandoci modo di poterli variare anche pagina per pagina in base al contenuto della medesima. Per come variare questi tag vi rimando ad altra documentazione sull'uso di Joomla in quanto è un argomento che non ha niente a che fare con i template.
Proseguendo ancora, incontriamo le seguenti righe:

```
<link rel="stylesheet" href="<?php echo $this->baseurl ?>/
templates/system/css/system.css" type="text/css" />

<link rel="stylesheet" href="<?php echo $this->baseurl ?>/
templates/system/css/general.css" type="text/css" />

<link rel="stylesheet" href="<?php echo $this->baseurl ?>/
templates/<?php echo $this->template ?>/css/template.css"
type="text/css" />
```

Chi abbia un minimo di conoscenza di html e css, capirà già che lo scopo di queste righe è chiamare in causa dei file css. La differenza sostanziale da un'altra comune pagina è la presenza di un paio di script php che sostituiscono parte del path che guida il browser verso il foglio di stile. Questo viene fatto per universalizzare il codice e renderlo valido per qualsiasi installazione di Joomla.
Se si osserva la riga:

```
<link rel="stylesheet" href="<?php echo $this->baseurl ?>/
templates/<?php echo $this->template ?>/css/template.css"
type="text/css" />
```

si può intuire come la parte di codice **<?php echo $this->baseurl ?>** indichi al browser la directory principale dell'installazione di joomla, mentre **<?php echo $this->template ?>** sta ad indicare il nome del template e quindi la directory che lo contiene.

Quindi, immaginando di avere un sito dal nome www.miosito.com all'interno del quale (nella direcotory principale del dominio) è installata una versione di joomla e che per visualizzarla stiamo usando un template di nome jomtemplate la stringa
<link rel="stylesheet" href="<?php echo $this->baseurl ?>/templates/<?php echo $this->template ?>/css/template.css" type="text/css" />

passerebbe al browser l'url

www.miosito.com/templates/jomtemplate/css/template.css omettendo però il nome di dominio (essendo una sorta di alias della directory root del dominio)

/templates/jomtemplate/css/template.css

Per essere più chiari, nel caso in cui l'istallazione di joomla si trovasse in una sottodirectory del dominio es.: www.miosito.com/sito/joomla la riga di cui sopra restituirebbe al browser il valore:

sito/joomla/templates/jomtemplate/css/template.css

Spero di essere stato chiaro in questo passaggio riguardante le path di un file, discorso molto più informatico che attinente a Joomla per il quale la cosa importante da comprendere è che con lo script

<?php echo $this->baseurl ?> si richiama la directory principale dell'istallazione di Joomla! Mentre con **<?php echo $this->template ?>** si richiama invece il nome della directory del template.

L'ultima cosa da anilizzare dell'head della pagina sono i file css linkati.
I file, come è facile capire sono tre:

template.css che è il file sul quale stiamo agendo e che contiene lo stile del template;
poi vi sono **general.css** e **system.css** che sono invece due file direttamente collegati al core di joomla e situati nella directory **/templates/system/css** dei quali però non ci occuperemo, sempre perché si tratta di qualcosa di più avanzato rispetto agli scopi di questa guida.

Quanto detto credo che sia più che sufficiente per darvi un'idea di come costruire un head della pagina index.php del vostro template, ma prima di sostituire il codice della pagina che abbiamo scritto sarà meglio approfondire cosa si dovrà invece fare nel body della pagina per far sì che questa si interfacci con Joomla.

Sia i moduli che i componenti devono essere richiamati all'interno del body del file index.php

[Dopo la testa c'è il corpo]

Dopo aver compreso come compilare un corretto head, è il momento di passare al corpo della pagina, dove si dovranno richiamare i vari "pezzi" di codice generati da Joomla.
Essendo Joomla un CMS che si appoggia ad un server database che contiene i contenuti veri e propri, questi non devono essere "scritti" all'interno della pagina ma bensì richiamati attraverso degli script.
Questo è un compito che si prende Joomla!, ciò che invece dobbiamo fare noi che creiamo templates è "istruire" Joomla! su dove far comparire i contenuti che volgiamo che compaiano, suddividendoli a nostro piacere negli spazi che abbiamo creato attraverso l'html e il css.
Per far sì che questo procedimento sia il più semplice possibile, gli sviluppatori di Joomla ci sono venuti incontro ideando quelle che possiamo definire come posizioni, lasciando liberi gli sviluppatori di template di ideare quante e più posizioni vogliono. Come creare queste posizioni lo vedremo nel capitolo dedicato al file templateDetails.xml, mentre per il momento utilizzeremo le posizioni "classiche" utilizzate dagli sviluppatori dei template di default di Joomla e che ci permetteranno di creare un template funzionante per un'istallazione di Joomla con i contenuti di esempio.
Le posizioni che utilizzeremo, per la fortuna di chi ci ha a che fare, hanno dei nomi dal suono molto "umano" e poco informatico. Quelle che vengono usate nel template di default Beez 2.0 sono:

debug
position-0
position-1
position-2
position-3
position-4
position-5
position-6
position-7
position-8
position-9
position-10
position-11
position-12
position-13
position-14

Come avrete capito smanettando un po' con Joomla che questo software è strutturato in maniera modulare, ovvero attraverso una serie di moduli, adempienti alle più svariate funzioni, che possiamo attivare o disabilitare, e che possiamo decidere, nel caso in cui li abilitiamo, di farli comparire in una qualsiasi posizione da noi scelta. Tutto questo è fattibile dal pannello di amministrazione, più precisamente dalla gestione moduli.

La modularità di Joomla però non si esaurisce nei moduli, essendo esso composto anche da plugin e componenti che però, per via della loro natura, non possono essere associati a delle posizioni. I plugin sono infatti delle estensioni in grado di aumentare le capacità di componenti e moduli, mentre i componenti sono le parti che gestiscono principalmente i contenuti del sito, come gli articoli per fare un esempio. Dei componenti ci occuperemo però in un secondo momento, metre ora ci concentreremo esclusivamente sui moduli.

< Inserire una posizione nel template >

Per essere il più possibile legati a situazioni reali e tangibili, affronteremo questo primo posizionamento di un modulo, prendendo come esempio il modulo main menu (menu principale, in Italiano), anch'esso gestibile, come tutti gli altri moduli, dalla gestione moduli del pannello di amministrazione di Joomla.
Ho scelto questo modulo in quanto è l'unico sempre attivato di default nelle installazioni di Joomla, siano esse fatte con i contenuti di esempio che senza. La posizione assegnata di default a questo modulo è la **position-7**.
Per far comparire questo modulo nel nostro template dovremo per prima cosa decidere dove andarlo ad inserire. Per quanto riguarda questa guida, ho scelto di inserirlo all'interno della colonna di sinistra.

Quindi, aprendo il file index.php che abbiamo creato in precedenza mi sposterò alla riga 33 dove compare:

```
<div id="colonna_sinistra">Colonna Sinistra</div>
```

e sostituire il contenuto "Colonna Sinistra" con uno script che richiami la posizione che ci interessa. Il risultato sarà il seguente:

```
<div id="colonna_sinistra">
<jdoc:include type="modules" name="position-7" style="XHTML" />
</div>
```

Come vedete la parte inserita **<jdoc:include type="modules" name="position-7" style="XHTML" />** non serve ad altro se non a passare a Joomla l'informazione di dover "stampare" all'interno del blocco div "colonna_sinistra" tutti i moduli caricati nella posizione di nome position-7, dandogli un output di tipo XHTML.

Nell'installazione di default di Joomla ci sono altri due moduli sempre attivati e sono:

Modulo Login
Modulo Percorso

Per default il modulo Login è assegnato anch'esso alla posizione position-7, mentre il modulo Percorso è assegnato alla posizione position-2

Andiamo quindi ad inserire la posizione position-2 all'interno del nostro template.

La posizione che ho scelto per caricare questo modulo è il div "pathway_container", quindi spostandoci nel file index.php dovremo andare a modificare la riga

```
<div id="pathway_container">Patway Breadcrumb</div>
```

in

```
<div id=”pathway_container”>
<jdoc:include type=”modules” name=”position-2” style=”XHTML” />
</div>
```

Come vedete il tutto è molto intuitivo e semplice da gestire. In sintesi esistono delle posizioni per il caricamento dei moduli richiamabili attraverso la stringa
<jdoc:include type=”modules” name=”nome-posizione” style=”tipo-di-output” />

Per quello che riguarda il nome delle posizioni, esse possono essere di completa fantasia, ma si badi bene che per funzionare devono essere precedentemente “dichiarate” all’interno del file **templateDetails.xml** (che vedremo in seguito)

Per quello che riguarda invece i diversi tipi di output, va detto che ne esistono molti, e che addirittura potrebbero esserne creati di nuovi, ma per il momento è sufficiente limitarsi allo stile: XHTML e allo stile non specificato, ovvero lasciando vuoto il valore style (es.: style=””)

< I Componenti >

I componenti sono quelle parti di Joomla che gestiscono alcuni contenuti come articoli, weblinks, feeds, contatti, banners ed altro ancora. Essi vengono sostanzialmente mostrati tutti nella stessa posizione, a differenza di quanto accade per i moduli.
La stringa da aggiungere al nostro file index.php al fine di richiamare i componenti è la seguente:

<jdoc:include type=”component” />

Come si può notare, questa stringa è molto simile a quella usata per i moduli, a differenza del fatto che in questo caso non esiste né un nome per la posizione, né una definizione dello stile di output. Usualmente si è soliti inserire la stringa **<jdoc:include type=”message” />** appena prima della stringa necessaria a richiamare i componenti, utile al sistema per la restituzione di messaggi o errori.

Questo capitolo tratta un argomento di natura avanzata
Rendere i moduli collassabili è una tecnica utile ma non certo “necessaria”

< Moduli collassabili >

Ciò che abbiamo fatto fino ad ora è stato semplicemente creare degli spazi nella pagina, attraverso del comune codice html e css, all’interno dei quali abbiamo posto delle stringhe in grado di richiamare componenti e moduli.
Come è ovvio, il template realizzato non è altro che una semplice bozza didattica per far comprendere il modo in cui Joomla stampa a video i contenuti all’interno della pagina del template. Quello che abbiamo fatto fino ad ora è però soltanto l’inizio del gioco. Non abbiamo ancora sfruttato che circa il 2% delle possibilità che invece avremmo.
Vediamo infatti adesso come potremmo decidere di far comparire e scomparire determinate parti della pagina, in base al fatto che all’interno di queste parti ci siano o meno dei contenuti .
Prendendo ad esempio una situazione abbastanza usuale, si potrebbe avere la necessità di rendere collassabile una colonna del template, per esempio la destra, che vorremmo scomparisse nel caso in cui al suo interno non ci fossero moduli attivi.
Vediamo come fare.

I codice di tutta la colonna è il seguente:

```
<div id=”colonna_destra”>Colonna Destra</div>
```

Al suo interno richiamiamo ora una posizione, diciamo la **position-10** e quindi avremmo:

```
<div id=”colonna_destra”>
<jdoc:include type=”modules” name=”position-10” style=”XHTML” />
</div>
```

Ora si dovrà in qualche modo far “reagire” la pagina in modo diverso nel caso in cui la position-10 sia attiva o meno. Per far questo ci viene in aiuto il php, attraverso un costrutto “if-then”.
Umanizzando il linguaggio, dovremmo scrivere del codice in grado di eseguire questa operazione: se la position-10 è attiva allora scrivi il codice della colonna, altrimenti saltalo a piè pari, che tradotto in php suona più o meno così:

```
<?php if($this->countModules(‘position-10’)) : ?>
     <div id=”colonna_destra”>
     <jdoc:include type=”modules” name=”position-10” style=”” />
     </div>
 <?php endif; ?>
```

La stringa **<?php if($this->countModules(‘position-10’)) : ?>** ha il compito di verificare un’opzione, che in questo caso coincide con la presenza o meno di un modulo attivo nella posizione **position-10**. Nel caso in cui questa opzione fosse verificata come positiva, la parte di codice contenuta entro la fine del costrutto <?php endif; ?>, verrebbe scritta sul file del template. In caso contrario, quindi con nessun modulo attivo in position-10, la parte di codice in questione (comprese le stringhe php) diverrebbe praticamente inesistente. Quello che è stato fatto per la colonna di destra del nostro template è però un’operazione incompleta. Abbiamo sì reso collassabile una parte del template, ma non ci siamo concentrati affatto sulle parti rimaste attive. La colonna centrale infatti, avendo uno stile con larghezza espressa in pixel, non subirà modificazioni nel caso in cui la colonna di destra sarà attivo o meno.

N.B.:Questa tecnica è solo uno dei tanti modi di rendere un modulo collassabile. Ne esistono anche altre!

Per realizzare un prodotto funzionale dovremmo quindi scrivere del codice che in qualche modo coinvolga anche la colonna centrale conseguenzialmente a ciò che succede per la colonna di destra.

Possiamo fare questo andando a creare delle variabili, sempre utilizzando del php, che si riempiranno di valori diversi a seconda che la colonna destra sia attivata o disattivata.
Il codice sottostante deve essere inserito tra i tag <head> ed </head> della pagina

```
<?php
   if($this->countModules('position-10') == 1) $infix = "_piccola";
   if($this->countModules('position-10') == 0) $infix = "_grande";
?>
```

Queste righe impostano il valore di una variabile di nome $infix, in base all'esistenza o meno di un modulo attivo in position-10. La prima riga, che corrisponde alla posizione attiva, dà alla variabile $infix il valore di "_piccola". La seconda riga invece, che corrisponde alla posizione non attiva, dà alla variabile $infix il valore di "_grande"
Spostandoci ora alla riga del file index.php dove viene richiamata la colonna centrale, ovvero:

```
<div id="colonna_centrale">
<jdoc:include type="component" /><
/div>
```

e modificandola in questo modo

```
<div id="colonna_centrale<?php echo ($infix)?>">
<jdoc:include type="component" />
</div>
```

avremo fatto sì che nel caso in cui ci fosse un modulo attivo in position-10, quindi la colonna destra attiva, allora il codice html richiamerebbe l'id colonna_centrale_piccola, mentre nel caso contrario si otterrebbe l'id colonna_centrale_grande.
Ora non si dovrà che apportare le dovute modifiche al file css per ottenere le due possibili colonne centrali, una più piccola (da usarsi quando la colonna destra è attivata) e una più grande (da usarsi invece quando la colonna destra è disattivata).

Quanto appena mostrato mette in evidenza come è possibile rendere estremamente dinamica la composizione di ogni pagina di un sito realizzato con Joomla. Creare un buon template significa dare la possibilità agli utenti di gestire la pagina in maniera più libera possibile, fornendo un prodotto adattabile a quante più esigenze possibile.

Il file templateDetails.xhtml è una parte fondamentale di ogni template!

[Il file templateDetails.xml]

Il file templateDetails.xml è una parte fondamentale di ogni template. Questo file gestisce tutto il funzionamento del template, dalla sua istallazione alle opzioni a questo applicabili. Le posizioni di cui abbiamo tanto parlato sono tutte espresse qui, come molti altri parametri. Vediamo la sua struttura:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE install PUBLIC "-//Joomla! 1.6//DTD template 1.0//EN"
"http://www.joomla.org/xml/dtd/1.6/template-install.dtd">
<extension version="2.5" type="template" client="site">
    <name>Nome</name>
    <creationDate>Data</creationDate>
    <author>Autore</author>
    <authorEmail>xxx@xxx.xx</authorEmail>
    <authorUrl>http://www.sitoautore.com</authorUrl>
    <copyright>Copyright (C) </copyright>
    <license>All Rights Reserved</license>
    <version>1.0.0</version>
    <description>My Template for my site</description>
```

Queste prime righe sono delle informazioni basilari, contenenti la versione di joomla per la quale è preparato il template, il nome del template, dell'autore, il copyright ecc...

Questa parte di codice <extension version="2.5" type="template" client="site"> è molto importante al fine di poter far installare automaticamente il template in una specifica versione di joomla. Fate molta attenzione a questa riga in quanto spesso si tende a dimenticarla (da come si vede nei forum) rendendo poi impossibile l'installazione del template.

Continuando nell'esplorazione di questo file troviamo le seguenti righe:

```
<files>
        <filename>index.php</filename>
        <filename>index.html</filename>
        <filename>templateDetails.xml</filename>
        <folder>images</folder>
        <folder>css</folder>

</files>
```

Come è possibile intuire, queste righe passano a Joomla un'informazione necessaria all'installazione del template, consistente nell'elenco dei file e delle directory che dovranno essere scritte sul disco. Nell'esempio proposto sono stati elencati soltanto i file e le cartelle fondamentali di un template, ma se ce ne dovessero essere delle altre, come una directory per i fonts, per fare un esempio, allora dovremmo inserire la riga:

```
<folder>fonts</folder>
```

[Il file templateDetails.xml]

Finito l'elenco dei file e delle cartelle, si passa all'elenco delle posizioni modulo da richiamare nel template. È qui che ogni posizione va elencata, pena la sua inutilizzabilità. Se una posizione non compare in questo elenco, non sarà mostrata all'utente qualora egli dovesse decidere di attivare o spostare un modulo dal backend del sito. È iportante dunque verificare che ogni posizione utilizzata nel file index.php sia anche elencata qui.
Ecco delle righe di esempio per mostrare come inserire una posizione nell'elenco:

```
<positions>

<position>debug</position>
<position>user1</position>
<position>user2</position>
<position>user3</position>
<position>user4</position>
<position>user5</position>
<position>user6</position>
<position>user7</position>
<position>user8</position>
<position>user9</position>
<position>user10</position>

</positions>
```

Come vedete l'elenco si apre e chiude con i tag <positions> e </positions> all'interno dei quali vengono caricate le voci, ognuna racchiusa tra i tag al singolare <position> e </position>

Superato questo elenco, si entra invece nella parte di file dedicata alle configurazioni e ai parametri modificabili del template. Questa è una parte del file che permette di aggiungere un'infinità di opzioni al template in costruzione, ma è qualcosa di troppo avanzato per questo tutorial. Limitatevi per il momento a saltarla, realizzando template più semplici ma comunque perfettamente funzionanti, lasciando vuota questa parte, scrivendo quindi solo la riga:

```
<config>
</config>
```

Il file si conclude qui, con la chiusura del tag <extension version="2.5" type="template" client="site"> che ovviamente sarà </extension>.
Ora il file templateDetails.xml è pronto all'uso.

CONCLUSIONI

Qui si conclude questa mini guida su come realizzare un template per Joomla. Non abbiamo affrontato argomenti avanzati ma credo che ci sia materiale sufficiente a darvi un'idea di come agire per creare un template per Joomla. Ora starà a voi darvi da fare e andare sempre avanti.

Buon divertimento!