

Title: Przewodnik wdrożeniowca > Praca z szablonami Flexy

Subject: eDokumenty - elektroniczny system obiegu dokumentów, workflow i CRM - DeployerGuide/Others/WorkingWithFlexy

Version: 78

Date: 06/05/25 20:57:36

Table of Contents

<i>Przewodnik wdrożeniowca > Praca z szablonami Flexy</i>	3
<i>Menu</i>	3
1. <i>Wprowadzenie</i>	3
2. <i>Tworzenie szablonu Flexy i wykorzystanie go w raportach</i>	3
2.1. <i>Tworzenie raportu SQL</i>	3
2.2. <i>Tworzenie szablonu Flexy dla raportu</i>	3
3. <i>Funkcje PHP dostępne w szablonach:</i>	5

[Przewodnik wdrożeniowca](#) > Praca z szablonami Flexy

Menu

1. [Wprowadzenie](#)
2. [Tworzenie szablonu Flexy i wykorzystanie go w raportach](#)
 - 2.1 [Tworzenie raportu SQL](#)
 - 2.2. [Tworzenie szablonu Flexy dla raportu](#)
3. [Funkcje PHP dostępne w szablonach](#)

1. Wprowadzenie

Flexy to bardzo szybki i dysponujący wielkimi możliwościami silnik szablonów. Może być używany zarówno w raportach, blokach jak i generowaniu plików zewnętrznych np. do połączenia z Symfonią.

<http://pear.php.net/manual/en/package.html.html-template-flexy.php>

Przejdź do [Menu](#)

2. Tworzenie szablonu Flexy i wykorzystanie go w raportach

2.1. Tworzenie raportu SQL

Tworzenie raportu z wykorzystaniem Flexy należy rozpocząć od zdefiniowania raportu SQL. W module raporty tworzymy nowy raport. W zakładkę *Ogólne* wprowadzamy nazwę raportu, miejsce jego przechowywania. Przechodzimy do zakładki *Definicja*, gdzie wprowadzamy definicję SQL. W zakładce Wybór kolumn możemy wybrać, jakie kolumny będą wykorzystywane w raporcie.

Przykładowo tworzymy raport dla *Notatek służbowych*. W tym celu tworzymy raport w grupie Dokumenty pt. *Notatka Służbowa*. W zakładce definicji raportu wprowadzamy SQL-a:

```
SELECT
dv.dscrpt,
dv.conten,
dv.srctxt,
dv.trgtxt,
p.symbol||' - '||p.dscrpt as symbol,
COALESCE(c.name_2, c.name_1) AS cname_
FROM documents_view dv
LEFT JOIN processes p USING (prc_id) --ON (dv.prc_id = p.prc_id)
LEFT JOIN doc_link_cont dlc ON (dv.doc_id = dlc.doc_id AND dlc.role__ = 'RELATED')
LEFT JOIN contacts c ON (dlc.contid = c.contid)
WHERE dv.doc_id = {DOC_ID}
```

Dodatkową funkcjonalnością, którą może być wykorzystana w raportach Flexy są kwerendy. Jest to ostatnia zakładka okienka definicji raportu. Parametry takiego raportu muszą być identyczne z parametrami głównego raportu. W odniesieniu do naszego przykładu mogą być to np. komentarze:

```
SELECT
dc.commnt,
u.firnam||' '|| u.lasnam as addtxt,
to_char(dc.addat, 'YYYY-MM-DD') AS addat
FROM document_comments AS dc
LEFT JOIN users u ON (dc.adduid = u.usr_id)
WHERE dc.doc_id = {DOC_ID} ORDER BY dc.addat ASC
```

Dodatkowy kwerend można zdefiniować dowolną ilość.

W tym momencie posiadamy działający już raport

2.2. Tworzenie szablonu Flexy dla raportu

Skoro posiadamy już raport SQL teraz należy utworzyć szablon. Czym jest szablon? Otóż jest to plik z rozszerzeniem **.html**, którego przykładowa zawartość wygląda następująco: {{{ <style>

```
TABLE .ReportTable TD {

    font-size: 12px;

}

TD.printLabel {

    font-size: 12px; vertical-align: top; font-style: italic; border-bottom: 1px solid grey;

}

TD.printData {

    font-size: 12px; vertical-align: top; border-bottom: 1px solid grey;

}

.printHeader {

    font-size: 16px; font-weight: bold; margin-bottom: 10px;

}

.parag {

    text-indent: 20px;

}

</style> <div style="margin-left:20px;">

    <p class="printHeader" style="margin-top: 10px;">Temat</p> {resultQueries[0][0][dscprt]}

    <p class="printHeader" style="margin-top: 10px;">Treść</p> {resultQueries[0][0][conten]:h}

    <p class="printHeader" style="margin-top: 10px;">Załączniki</p> <ul style="list-style-type:none;">

        {foreach:resultQueries[2],k,v}

            <li style="font-size:12px;"> - {v[filenm]}</li>

        {end;}

    </ul>

    <p class="printHeader" style="margin-top: 10px;">Komentarze</p> <ul style="list-style-type:none;">

        {foreach:resultQueries[1],k,v}

            <li><i>{v[addtxt]} {v[adddat]}</i><BR>{v[commnt]:h}</li>

        {end;}

    </ul>

    <table class="RaportTable RaportTable2" style="margin-top:20px; background-color: white; width: 100%" border=0 cellpadding=1 cellspacing=2>

        <tr>

            <td class="printLabel">Dokument od:</td> <td class="printData">{resultQueries[0][0][srctxt]}</td> <td class="printLabel">Dokument do:</td> <td class="printData">{resultQueries[0][0][trgtxt]}</td>

        </tr> <tr>
```

```
<td class="printLabel">Klient:</td> <td class="printData">{resultQueries[0][0][cname_]}</td> <td class="printLabel">Numer  
sprawy:</td> <td class="printData">{resultQueries[0][0][symbol]}</td>
```

```
</tr>
```

```
</table>
```

```
</div> }}}
```

Należy zwrócić uwagę, że nie ma tutaj definicji typowych dla pliku html, jak `<html>`, `<head>`, `<body>`, `<!DOCTYPE...>`. Jest niejako tylko czysta treść.

Przejdź do [Menu](#)

3. Funkcje PHP dostępne w szablonach:

1. str_replace

Przykład użycia:

```
{str_replace(#-#, ##, documents.dscript):h}
```

Powyższe wywołanie usunie wszystkie wystąpienia znaku "-" z tekstu znajdującego się pod zmienną "documents.dscript"

1. trim

```
{trim(documents.dscript):h}
```

Powyższe wywołanie usunie wszystkie białe znaki z początku i końca tekstu znajdującego się pod zmienną "documents.dscript"

1. preg_replace

1. substr

```
{substr(documents.dscript, 2, 0):h}
```

Powyższe wywołanie zwróci tekst ze zmiennej "documents.dscript" począwszy od drugiego znaku.

1. strpos / stripos

```
Opis dokumentu {if:strpos(documents.dscript, #123#)}ZAWIERA{else:}NIE ZAWIERA{end:} ciąg: "123"
```

1. funkcje porównujące (eq, gt, gte, lt, lte)

```
eq(1,2) jednoznaczne z 1 == 2  
gt(1,2) jednoznaczne z 1 > 2  
gte(1,2) jednoznaczne z 1 >= 2  
lt(1,2) jednoznaczne z 1 < 2  
lte(1,2) jednoznaczne z 1 <= 2
```

przykład:

```
{if:gt(vatnote.netto_,0)}  
kwota netto jest większa od zera  
{end:}
```

wszystkie dostępne funkcje:

```
round($val, $precision = NULL);
```

```
in_array($val, $array);

array_key_exists($key, $array);

strpos($haystack, $needle);

stripos($haystack, $needle);

substr($string, $start, $length);

str_replace($from, $to, $source);

trim($string);

preg_quote($str);

preg_replace($from, $to, $source);

date($format, $time = NULL);

number_format( float $number , int $decimals , string $dec_point , string $thousands_sep )

str_pad( string $input , int $pad_length [, string $pad_string= " " [, int $pad_type= STR_PAD_RIGHT ]] )
```

Przejdź do [Menu](#)