Step 2 - Display the list of Rides on the homepage

- 1 Custom your homepage template
- 2 Create your sub controller to display list of Rides
- 3 Create template to display the list of Rides
- 4 Use a custom template to display view line of a Ride

Tutorial path

Custom your homepage template

Let's modify the `Resources/views/content/full/root_folder.html.twig` adding a call to a subrequest to display the list of all existing Rides with pagination:

root_folder.html.twig

```
{% extends "pagelayout.html.twig" %}

{% block content %}
    <script src="http://maps.google.com/maps/api/js?sensor=false"></script>

    <h3 class="center bottom-plus new-header">{{ ez_content_name(content) }}</h3>
    <div class="col-xs-10 text-justified">
    {{ render( controller( "AppBundle:Homepage:getAllRides" ) ) }}

{% endblock %}
```

For the moment, we use a simple `render()` Twig function but when we talk about cache, we will use **`render_esi`**.

Create your sub controller to display list of Rides

Create your `/src/AppBundle/Controller/HomepageController.php` with the function `getAllRidesAction`:

HomepageController.php

```
<?php
namespace AppBundle\Controller;
use eZ\Publish\API\Repository\Values\Content\Query;
use eZ\Publish\API\Repository\Values\Content\Query\Criterion;
use eZ\Publish\Core\MVC\Symfony\Controller\Controller;
use eZ\Publish\API\Repository\Values\Content\Query\SortClause;
use eZ\Publish\Core\Pagination\Pagerfanta\ContentSearchAdapter;
use Pagerfanta\Pagerfanta;
use Symfony\Component\HttpFoundation\Request;
use eZ\Publish\API\Repository\Values\Content\Location;
use
eZ\Publish\API\Repository\Values\Content\LocationQuery;
class HomepageController extends Controller
{
    public function getAllRidesAction(Request $request)
    {
        $repository = $this->getRepository();
        $locationService =
$repository->getLocationService();
        $contentService =
$repository->getContentService();
        $rootLocationId =
$this->getConfigResolver()->getParameter('content.tree_r
oot.location_id');
        $rootLocation =
$locationService->loadLocation($rootLocationId);
        $currentLocationId = 2;
        $currentLocation =
$locationService->loadLocation($currentLocationId);
        $currentContent =
$contentService->loadContentByContentInfo($currentLocati
on->contentInfo);
        $query = new Query();
        $query->query = new Criterion\LogicalAnd(
            array(
                new
Criterion\Subtree($rootLocation->pathString),
                new
Criterion\Visibility(Criterion\Visibility::VISIBLE),
                new
Criterion\ContentTypeIdentifier(array('ride'))
            )
        );
        $query->sortClauses = array(
            new
SortClause\DatePublished(Query::SORT_ASC)
        );
        $pager = new Pagerfanta(
            new ContentSearchAdapter($query,
$this->getRepository()->getSearchService())
        );
        //FIXME : get $limit value from a custom
parameter
        $limit = 10;
        $pager->setMaxPerPage($limit);
        $pager->setCurrentPage($request->get('page',
1));
        return $this->render(
            'content/view/list/rides.html.twig',
            array(
                'location' => $currentLocation,
                'content' => $currentContent,
                'pagerRides' => $pager,
            )
        );
    }
}
```

Create template to display the list of Rides

Create `app/Resources/AppBundle/views/list/rides.html.twig` template. You use a <table> to display the list of rides. The <thead> of the <table> is in this Ride list template and each <tr> (line of the table) is in the line ride template.

So each time you will use the line Ride template, you have to remember the choice of using a <tr>.

rides.html.twig

```
{#Only display pagerfanta navigator if needed.#}
{% if pagerRides.haveToPaginate() %}
<nav class="text-center">
    <ul class="pagination">
    <div class="pagerfanta">
        {{ pagerfanta( pagerRides,
'twitter_bootstrap_translated', {'routeName': location }
) }}
    </div>
    </ul>
</nav>
{% endif %}
<div class="row regular-content-size">
    <div class="col-md-8 col-md-offset-2 box-style">
        <h3 class="center bottom-plus new-header">{{
'List of all Rides' | trans }}</h3>
        {# Loop over the page results #}
        {% for ride in pagerRides %}
            {% if loop.first %}
                <table class="table table-hover">
                <thead>
                <tr class="table-header">
                    <th></th>
                    <th>Name</th>
                    <th>Starting Point</th>
                    <th>Ending Point</th>
                    <th>Length</th>
                    <th>Level</th>
                </tr>
                </thead>
                <tbody>
            {% endif %}
            {{ render( controller(
'ez_content:viewLocation', { 'locationId':
ride.versionInfo.contentInfo.mainLocationId, 'viewType':
'line' } )) }}
            {% if loop.last %}
                </tbody>
                </table>
            {% endif %}
        {% endfor %}
    </div>
</div>
{#Only display pagerfanta navigator if needed.#}
{% if pagerRides.haveToPaginate() %}
    <nav class="text-center">
        <ul class="pagination">
            <div class="pagerfanta">
                {{ pagerfanta( pagerRides,
'twitter_bootstrap_translated', {'routeName': location }
) }}
            </div>
        </ul>
    </nav>
{% endif %}
```

The next step is to create the override rule to use a dedicated template for the view line of Rides.

To do so, you need to configure your bundle to inject override configuration.

Use a custom template to display view line of a Ride

You add the rule for the line_ride template to be used in your `app/config/ezplatform.yml` file.

ezplatform.yml

```
system:
    site_group:
        content_view:
            line:
                line_ride:
                    template:
"content/view/line/ride.html.twig"
                    match:
                        Identifier\ContentType: "ride"
```

Create your `app/Resources/AppBundle/views/content/view/line/ride.html.twig` template. Remember, it's only one line of a table, so you will find a <tr> tag with some <td> tags.

ride.html.twig

```
<tr>
    <td>
        {{ ez_render_field( content, 'image', {
parameters: { 'alias': 'small' }} ) }}
    </td>
    <td>
        <strong>
            <a href="{{ path( "ez_urlalias", {
'locationId': content.contentInfo.mainLocationId } ) }}"
               target="_self">
                {{ ez_content_name( content ) }}
            </a>
        </strong>
        <p class="pre-small">
            {{ ez_render_field( content, 'author') }}
        </p>
    </td>
    <td>
        {{ ez_render_field(content, 'starting_point',
{'parameters': {'width': '100%', 'height': '100px',
'showMap': true, 'showInfo': true }}
        ) }}
    </td>
    <td>
        {{ ez_render_field(content, 'ending_point',
{'parameters': {'width': '100%', 'height': '100px',
'showMap': true, 'showInfo': true }}
        ) }}
    </td>
    <td>
        <p>{{ ez_render_field( content, 'length' ) }}
Km</p>
    </td>
    <td>
        <p>{{ ez_render_field( content, 'level' ) }}</p>
    </td>
</tr>
```

Previous: Step 1 - Display content of a Ride

Next : Congrats!