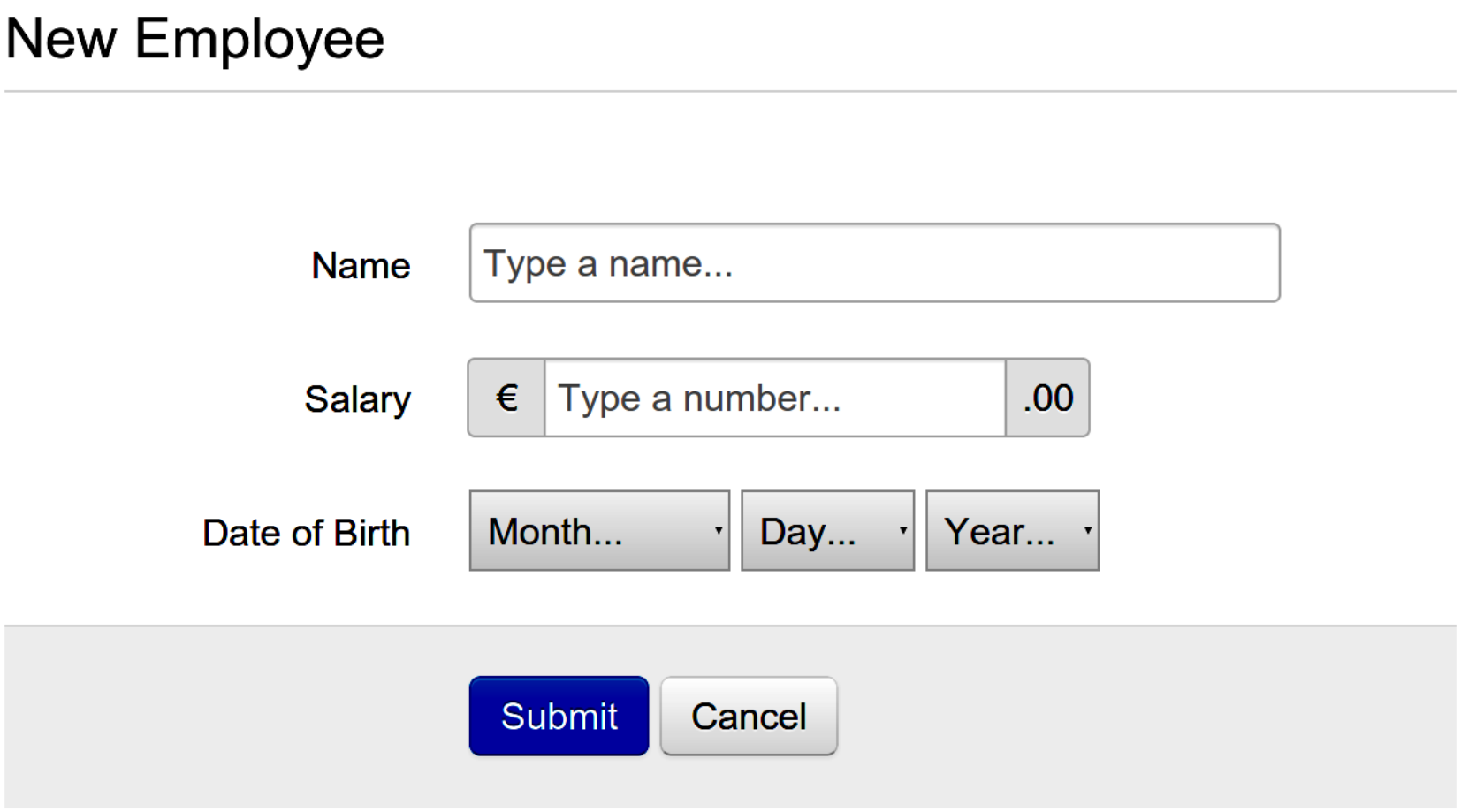

Symfony2 Form Tricks

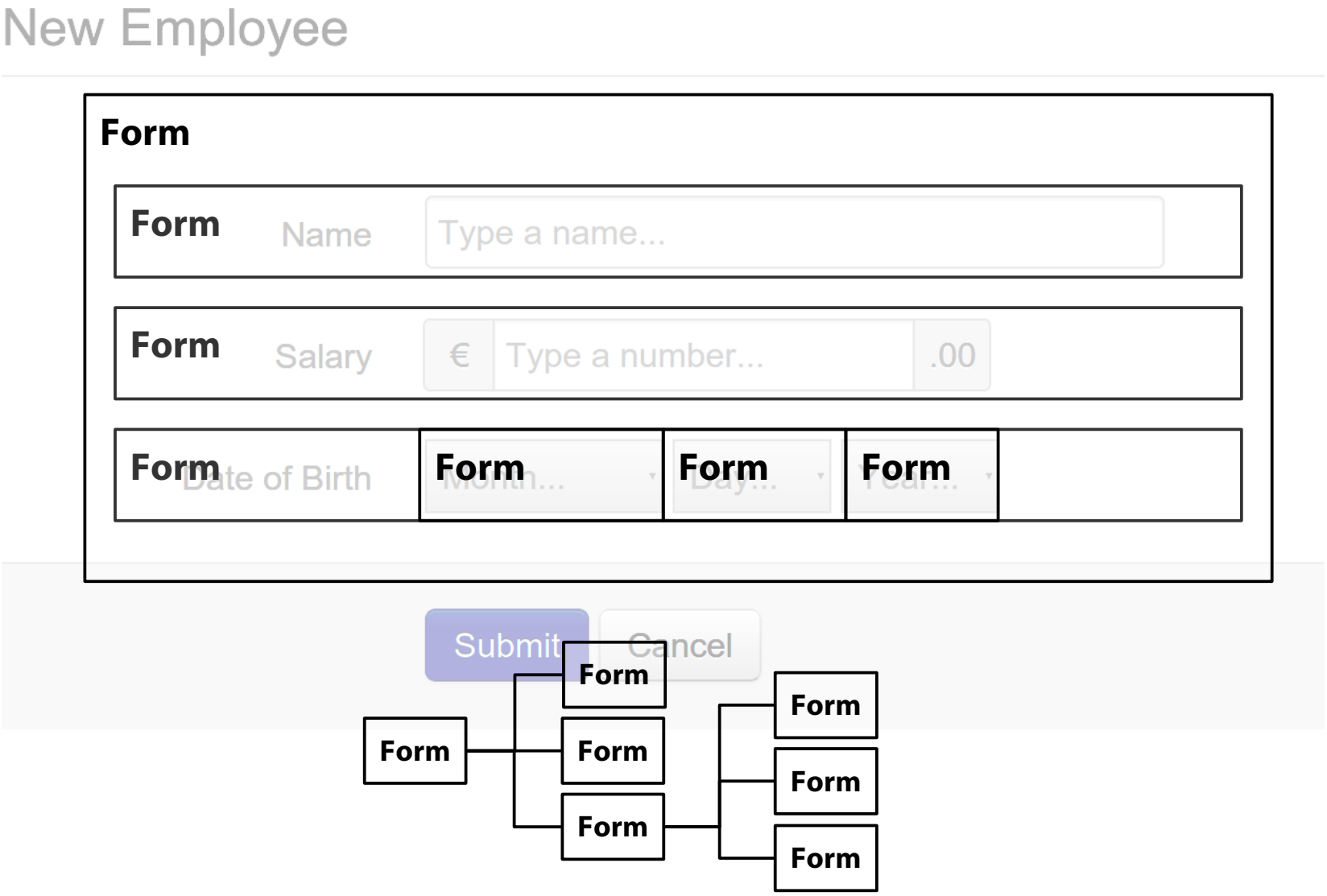
Outline

- Introduction
- Data formats
- Data mapping
- Events

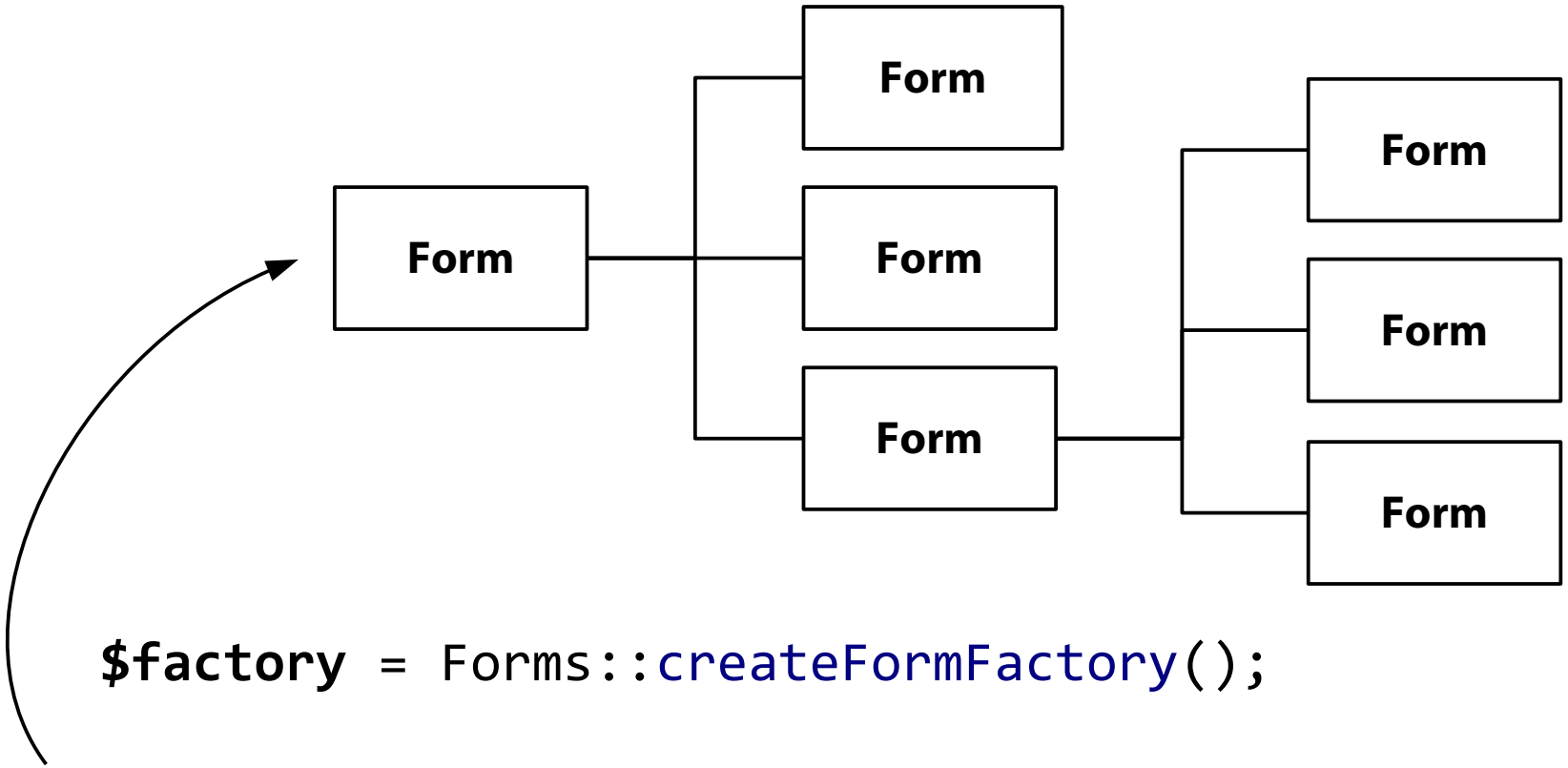
Today's Example



Symfony's Perspective

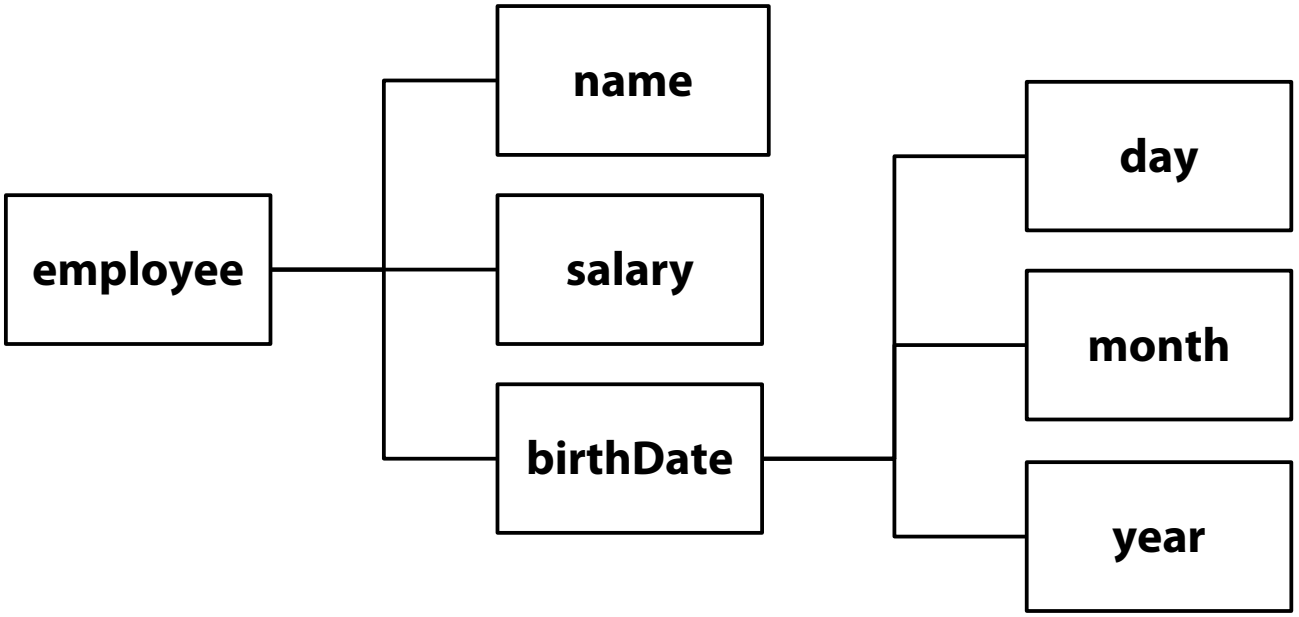


Form Tree



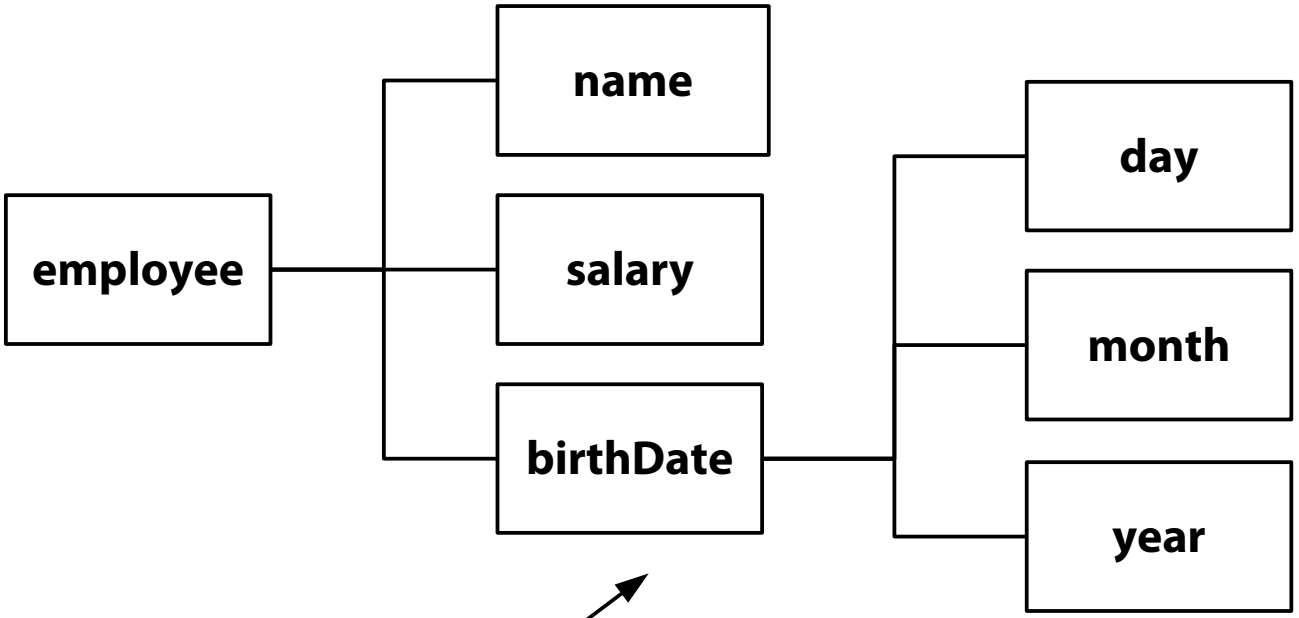

```
$factory = Forms::createFormFactory();

$form = $factory->createForm(...);
```

Forms Have Names



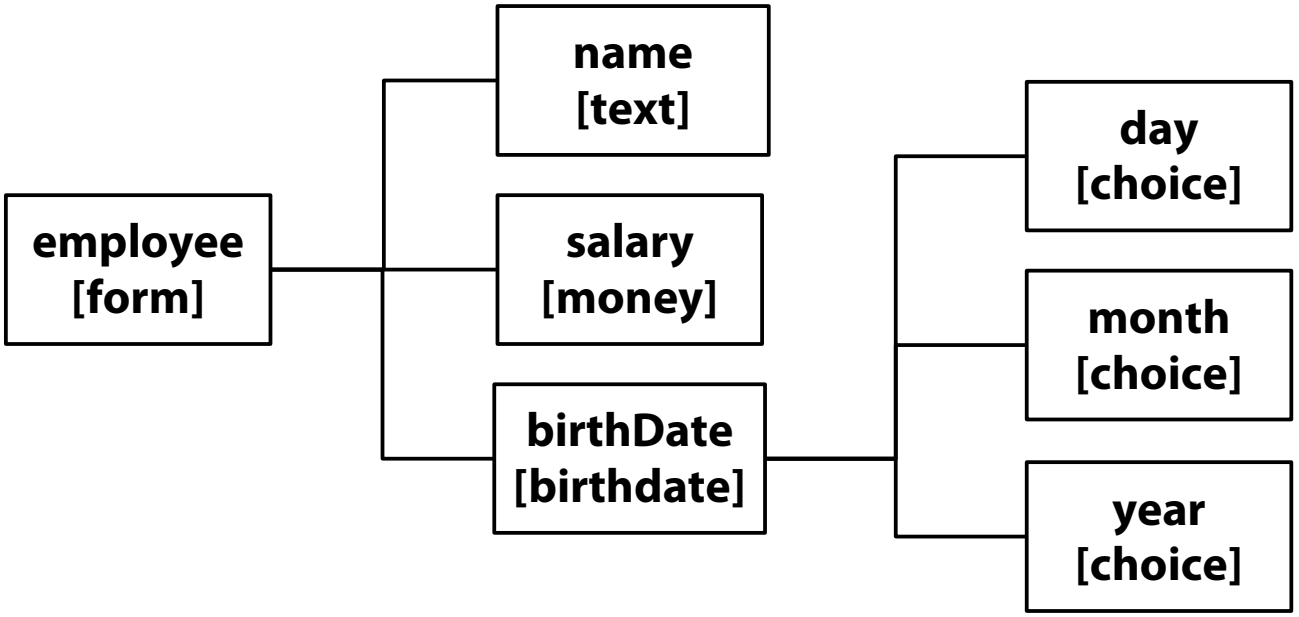

```
$builder = $factory->createNamedBuilder('employee');

$builder->add('name');
$builder->add('salary');
$builder->add('birthDate');

$form = $builder->createForm();
```

Forms Have Names



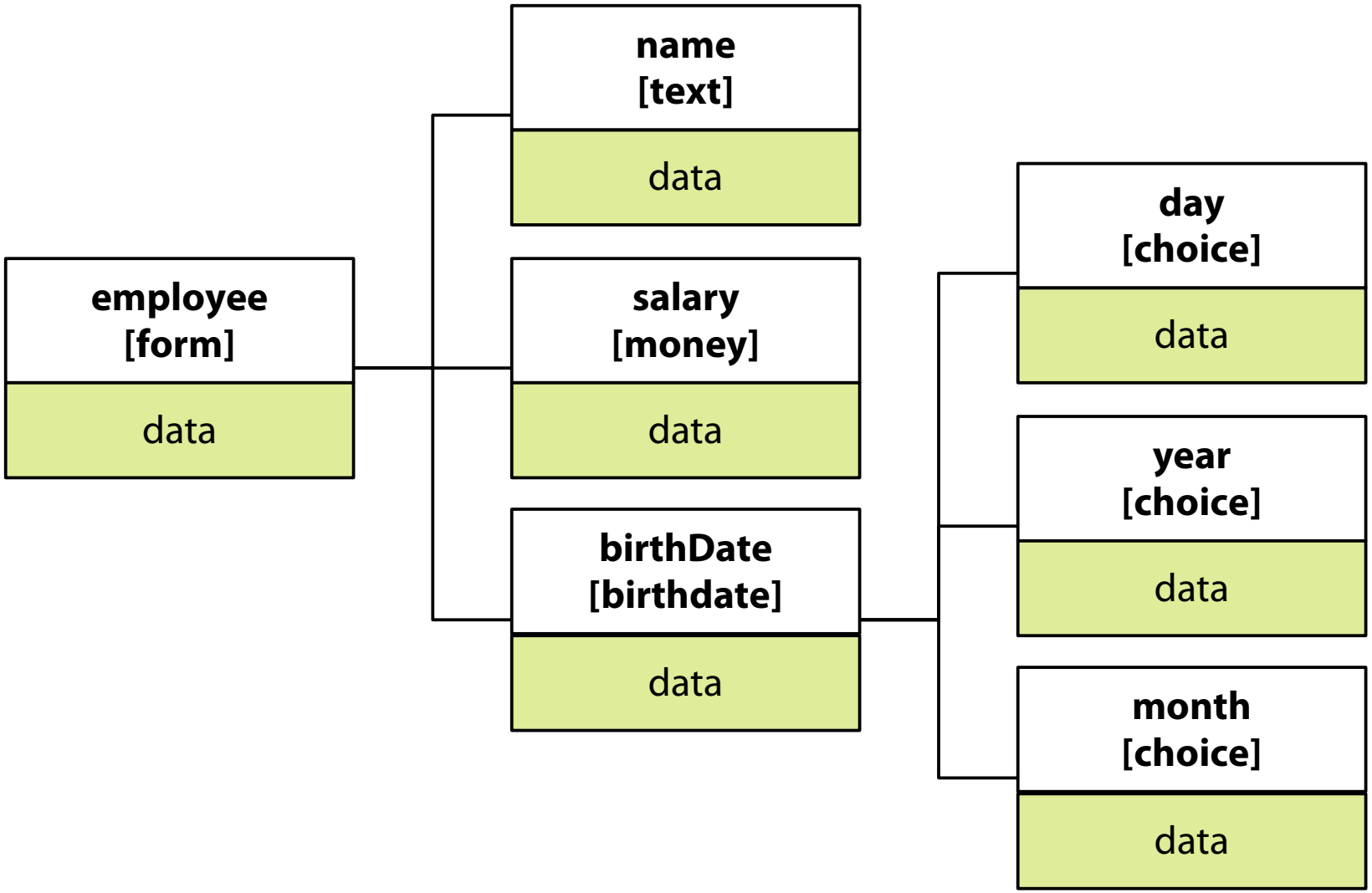

```
$form->get('birthDate');

$form->get('birthDate')->get('year');
```

Forms Have Types

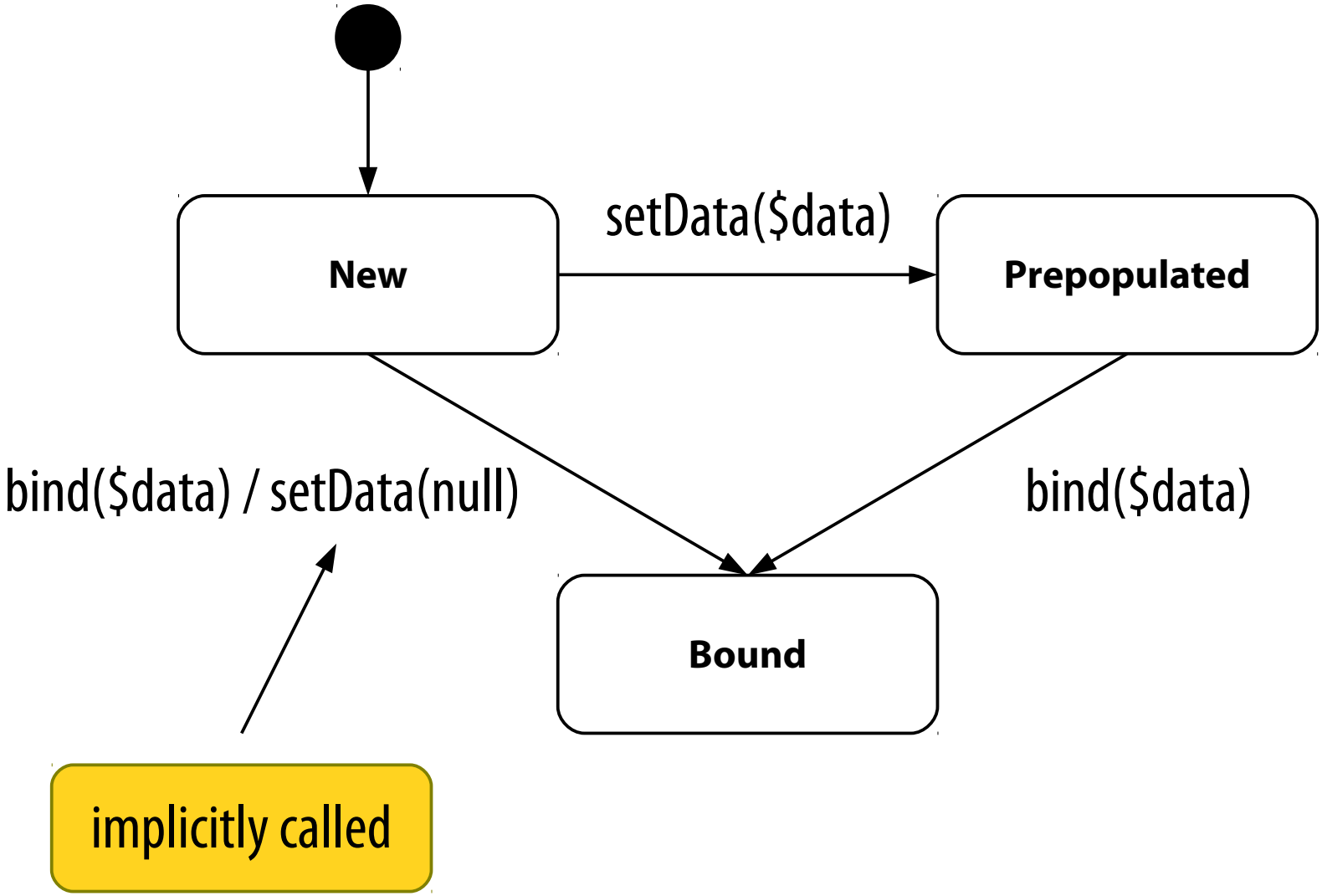



```
$builder = $factory->createNamedBuilder('employee', 'form');

$builder->add('name', 'text');
$builder->add('salary', 'money');
$builder->add('birthDate', 'birthdate');

$form = $builder->createForm();
```

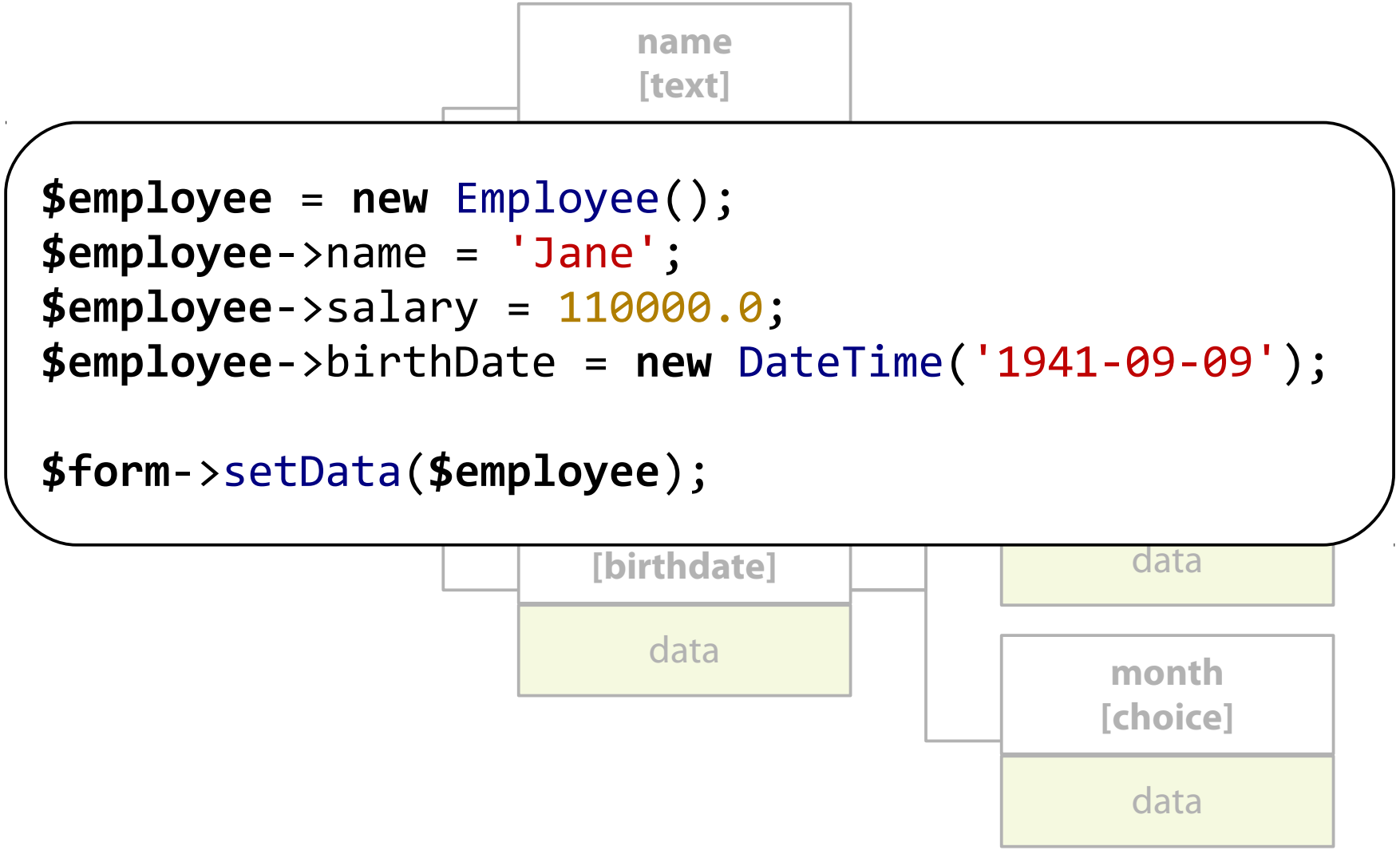
Forms Have Data



Form Lifecycle

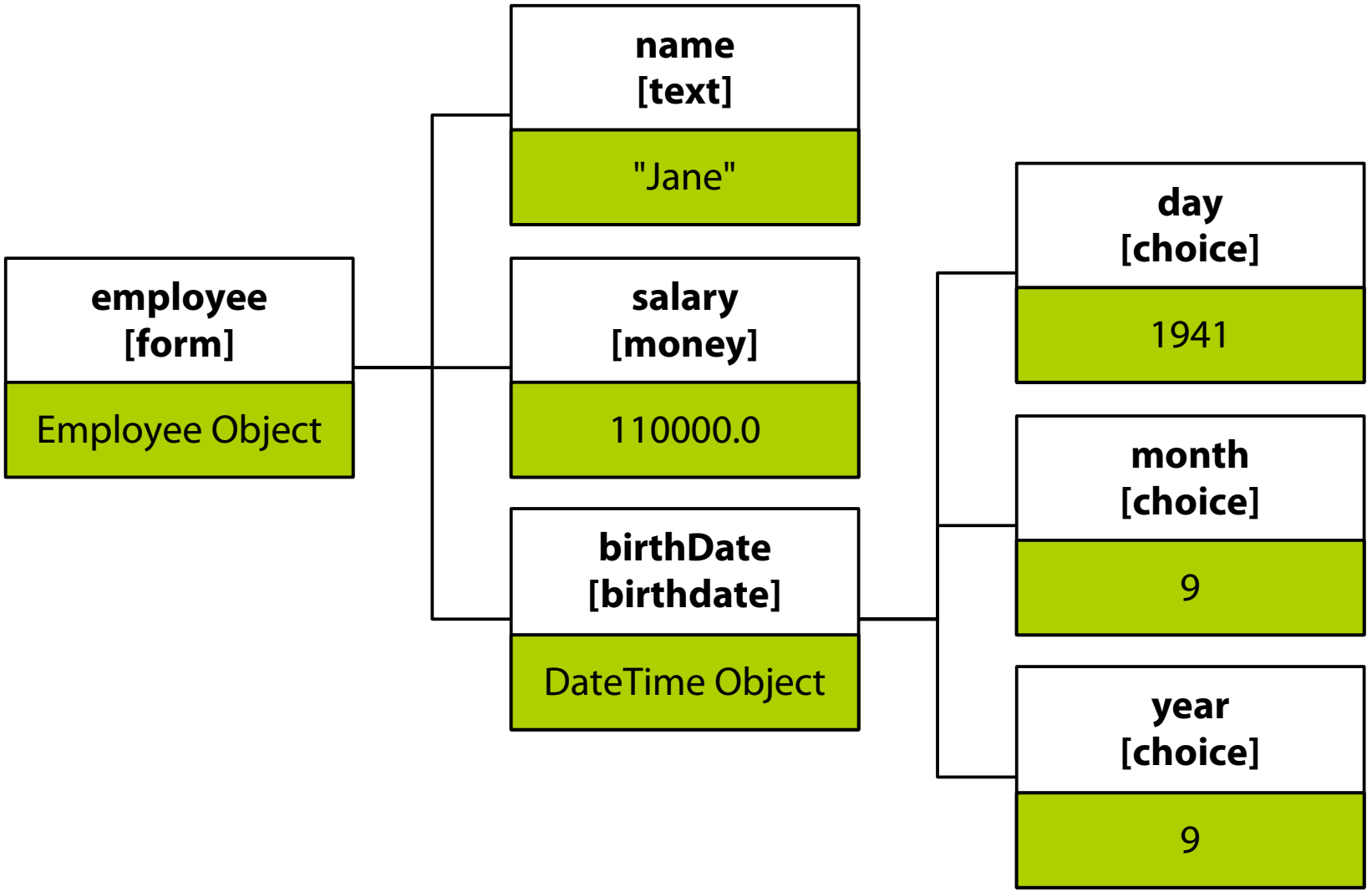


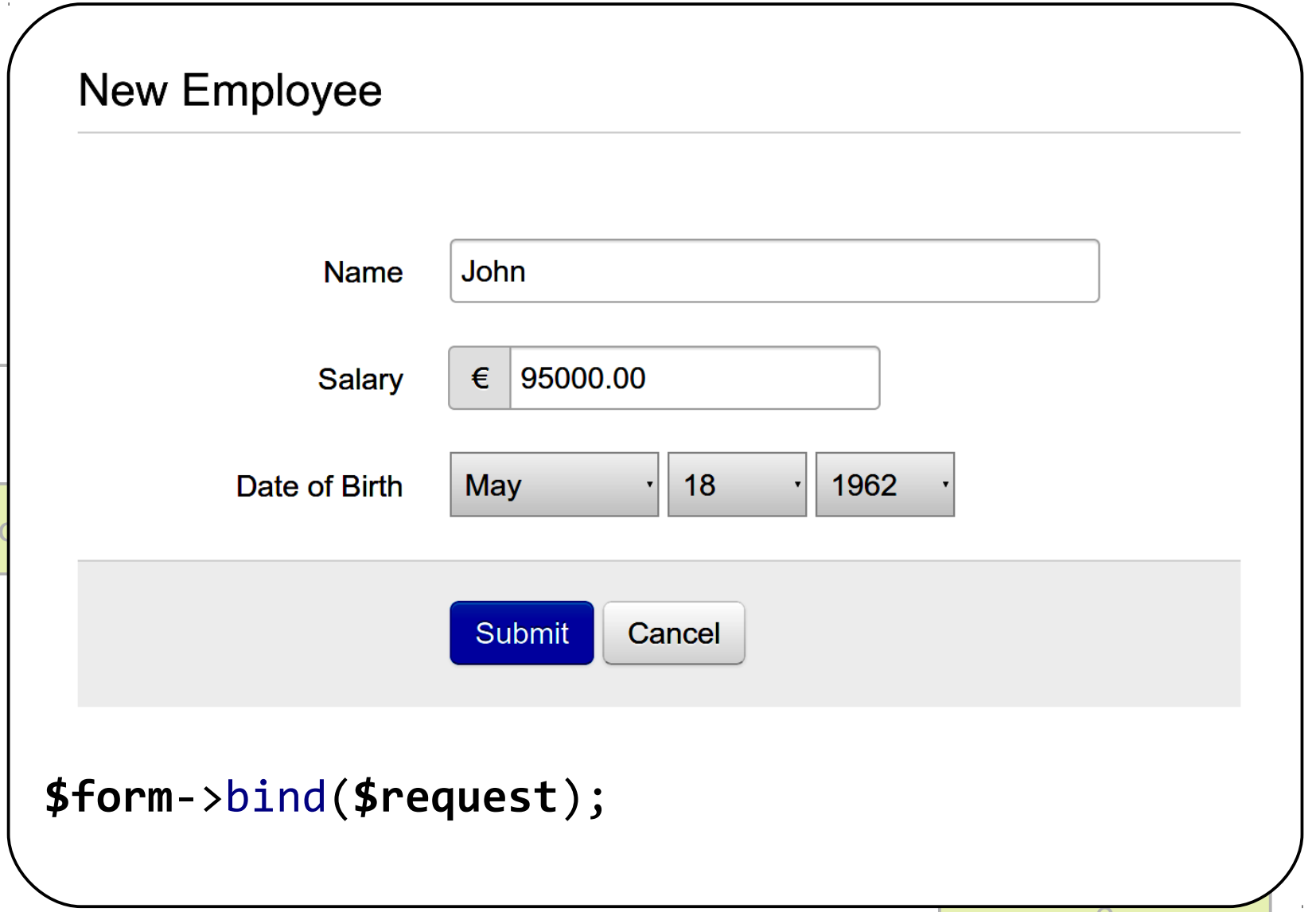
Prepopulation



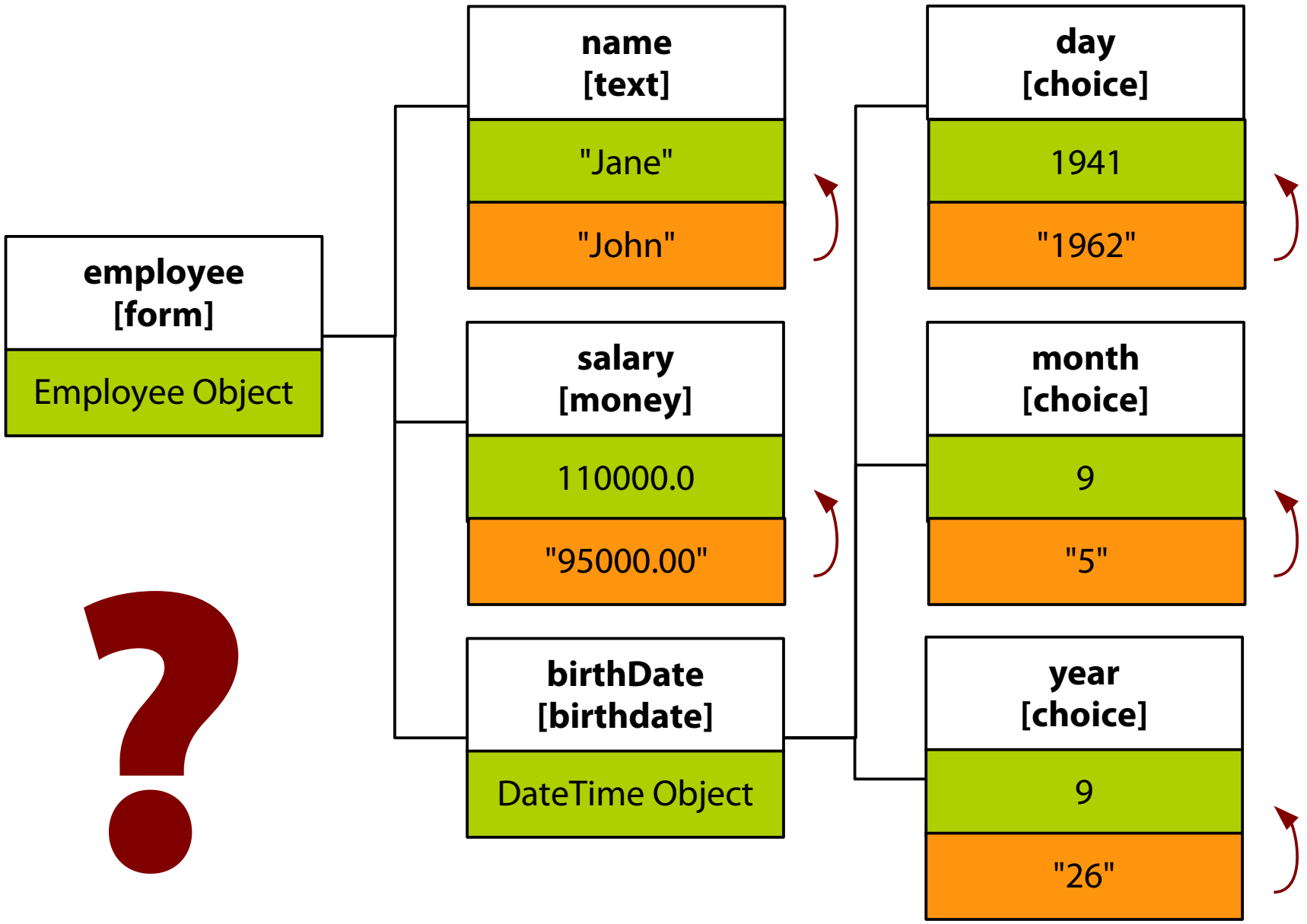

```php
$employee = new Employee();
$employee->name = 'Jane';
$employee->salary = 110000.0;
$employee->birthDate = new DateTime('1941-09-09');

$form->setData($employee);
```

Prepopulation

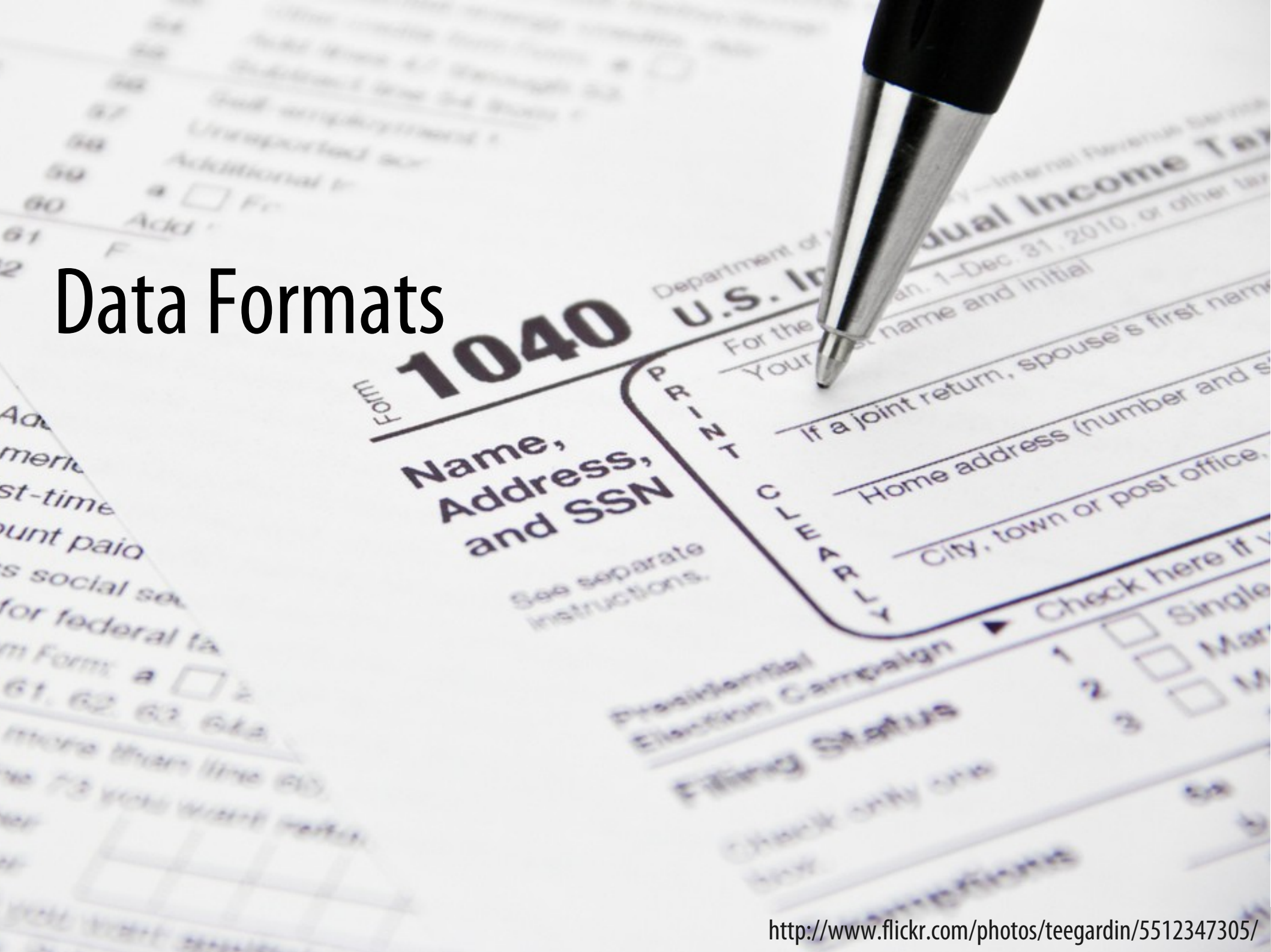





```
$form->bind($request);
```

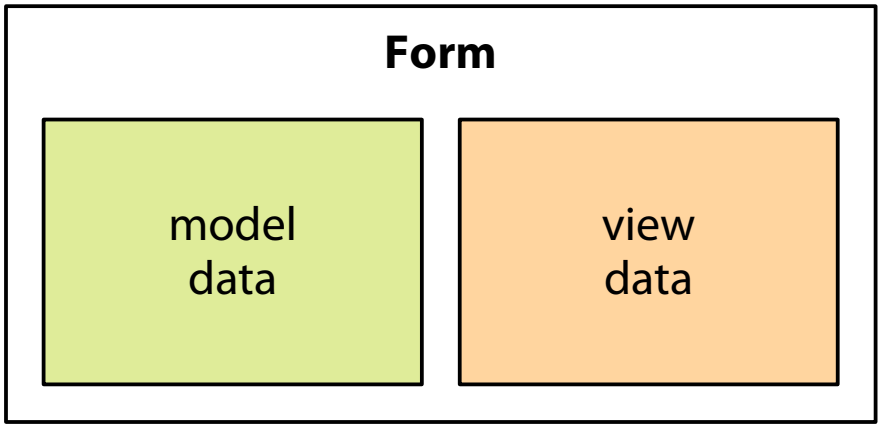

employee
[form]
Employee Object
name
[text]
"Jane"
"John"
salary
[money]
110000.0
"95000.00"
birthDate
[birthdate]
DateTime Object
day
[choice]
1941
"1962"
month
[choice]
9
"5"
year
[choice]
9
"26"
?

Data Formats

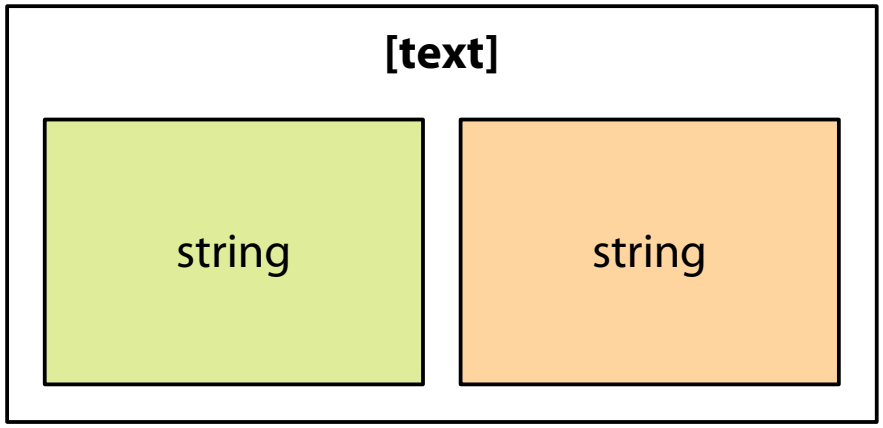


http://www.flickr.com/photos/teegardin/5512347305/

Data Formats

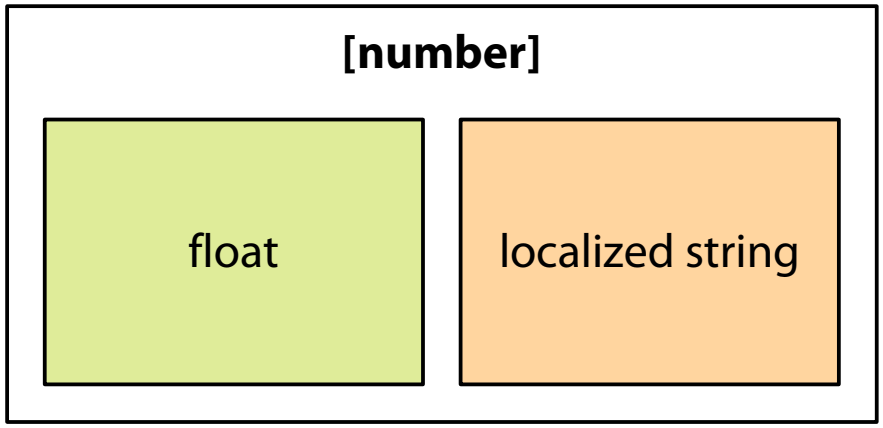


Examples



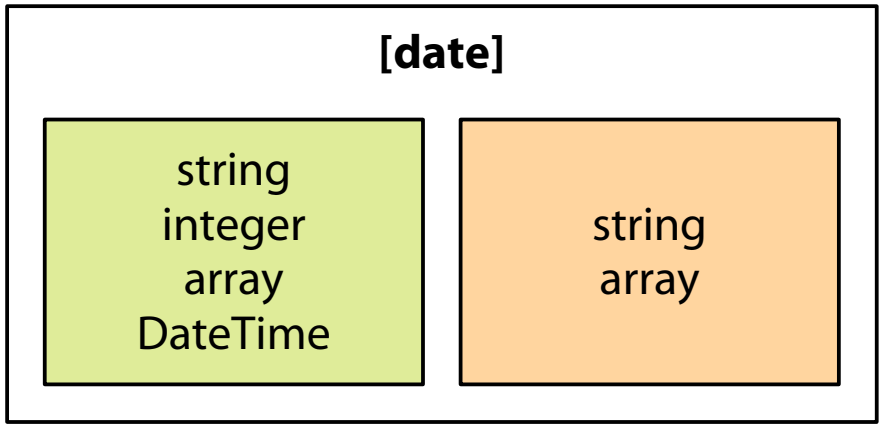
Bernhard

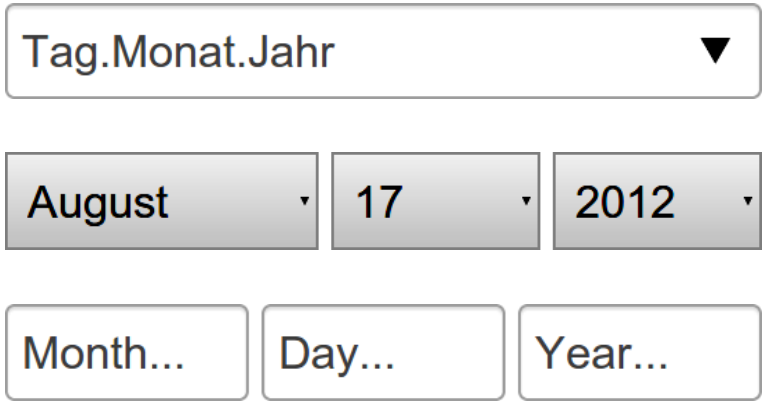
Examples



100.000,00

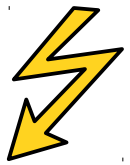
Examples



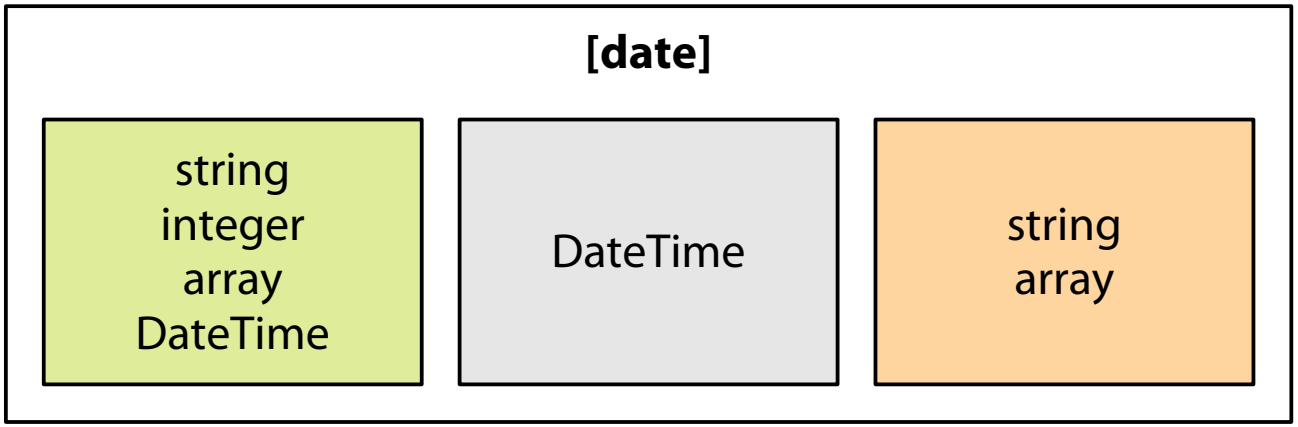



```
$builder->add('createdAt', 'date', array(
    'input'  => 'array',
    'widget' => 'single_text',
));
```

```php
class MyDateType extends AbstractType
{
    public function getParent()
    {
        return 'date';
    }

    public function buildForm(FormBuilderInterface $builder)
    {
        $builder->addEventListener(FormEvents::BIND,
            function (FormEvent $event) {
                // which format do I get???
                $data = $event->getForm()->getData();
            }
        );
    }
}
```



Normalized Format



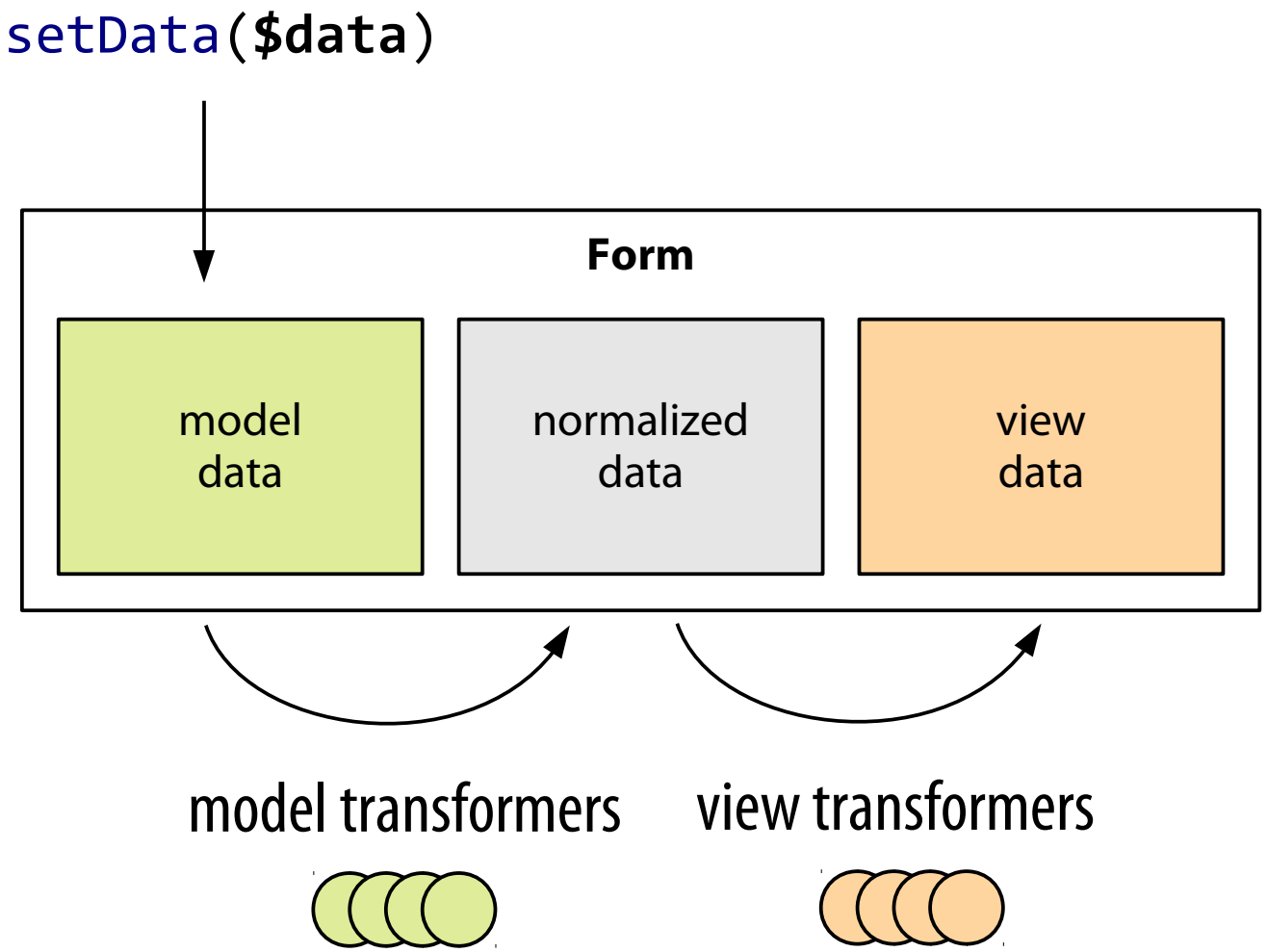

```
$builder->addEventListener(FormEvents::BIND,
    function (FormEvent $event) {
        $data = $event->getForm()->getNormData();
    }
);
```

Tip 

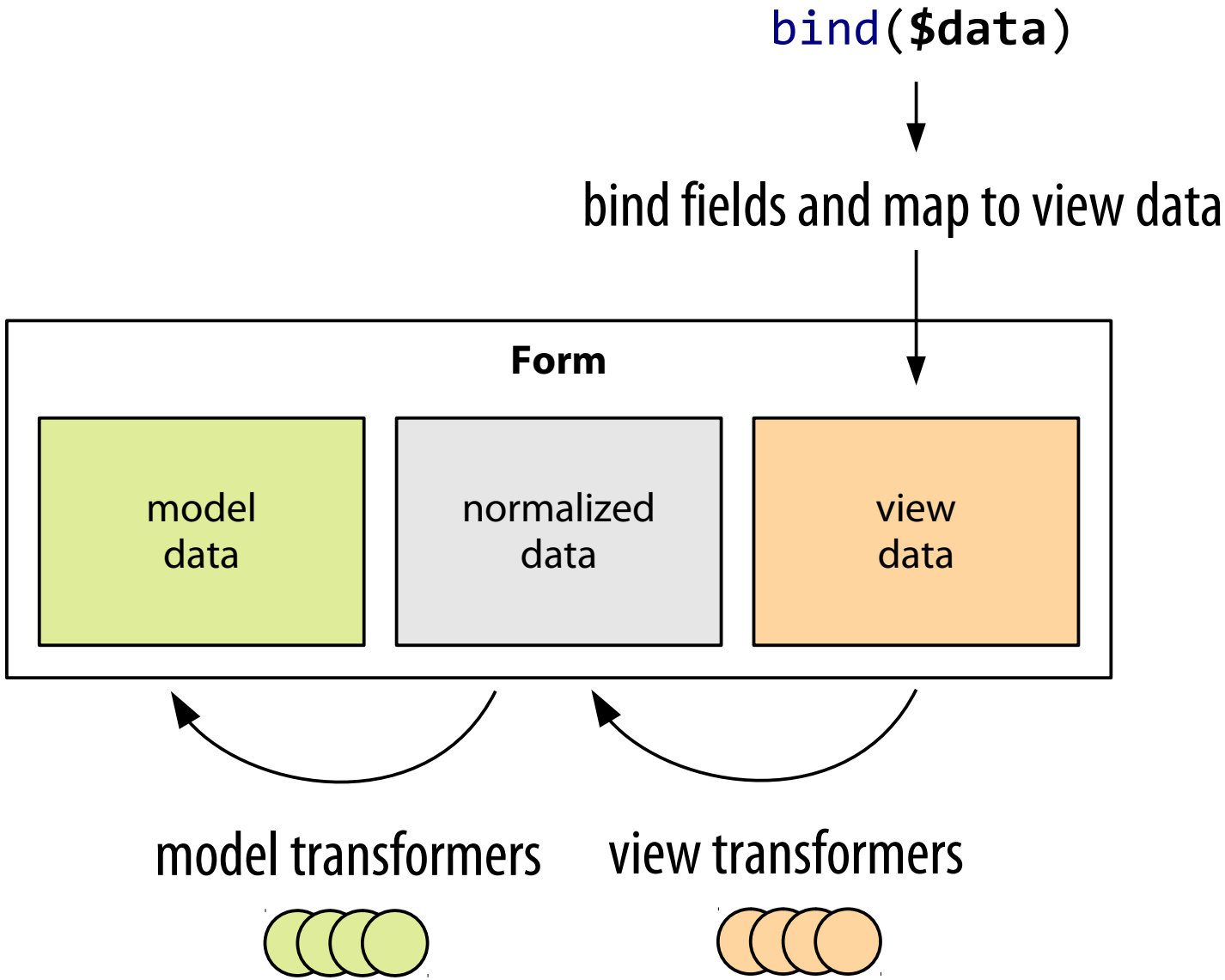
- Whenever you create a custom type, decide:
  - Model format?
  - Normalized format?
  - View format?

http://www.flickr.com/photos/xverges/4622486563/

Data Transformation



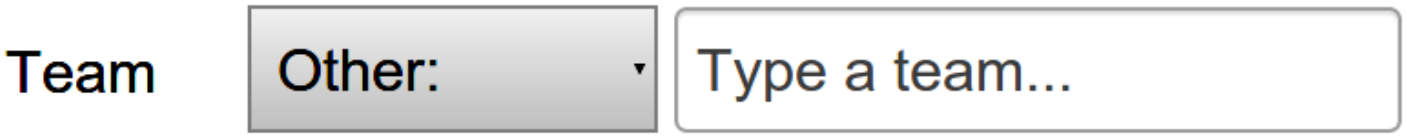
Data Transformation



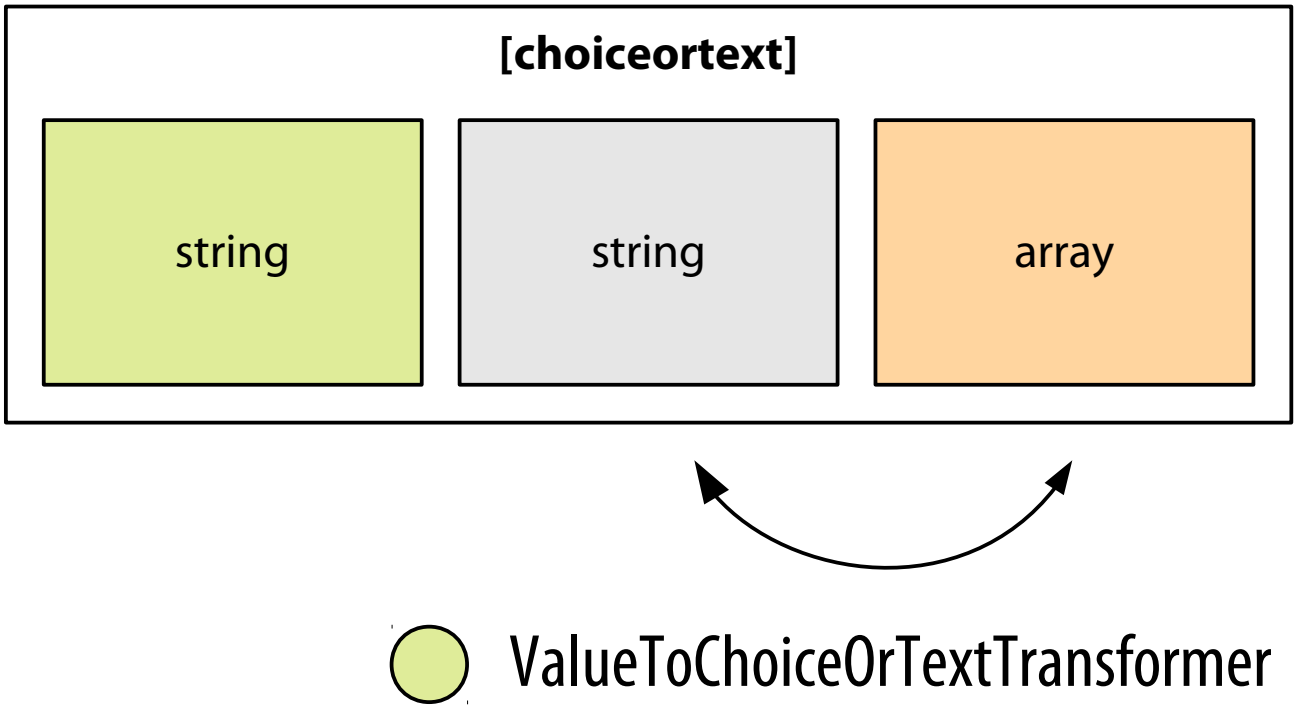
Data Transformation

- Symfony\Component\Form\DataTransformerInterface
  - bijective
  - transform($data)
  - reverseTransform($data)

Example: ChoiceOrTextType



Example: ChoiceOrTextType

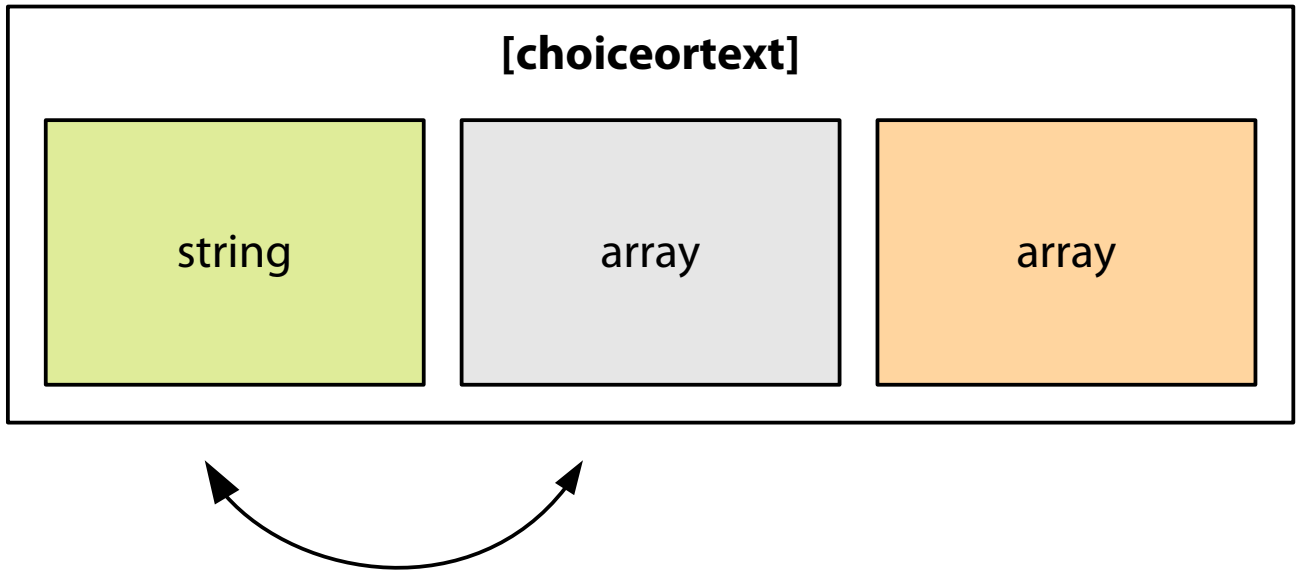


Tip 

The normalized data should contain as much information as possible.

http://www.flickr.com/photos/xverges/4622486563/

Example: ChoiceOrTextType

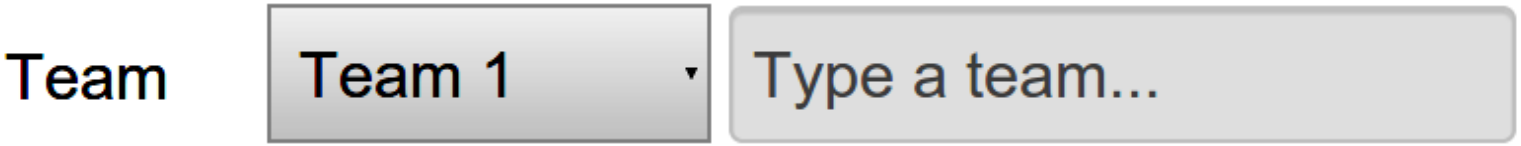


ValueToChoiceOrTextTransformer

Transformer Implementation

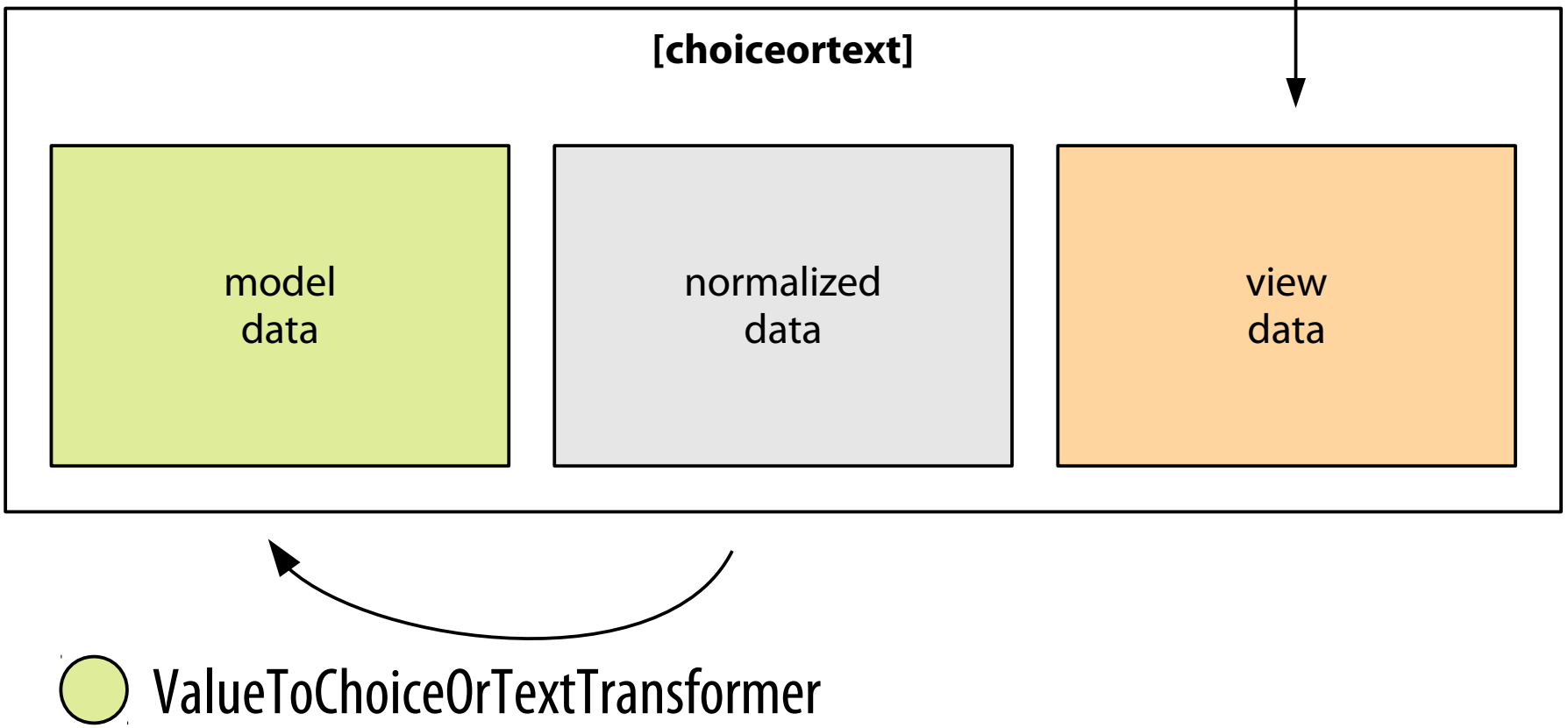
```
use Symfony\Component\Form\DataTransformerInterface;

class ValueToChoiceOrTextTransformer implements
        DataTransformerInterface
{
    public function transform($data)
    {
        ...
    }

    public function reverseTransform($data)
    {
        ...
    }
}
```

Case 1: Existing Team Selected



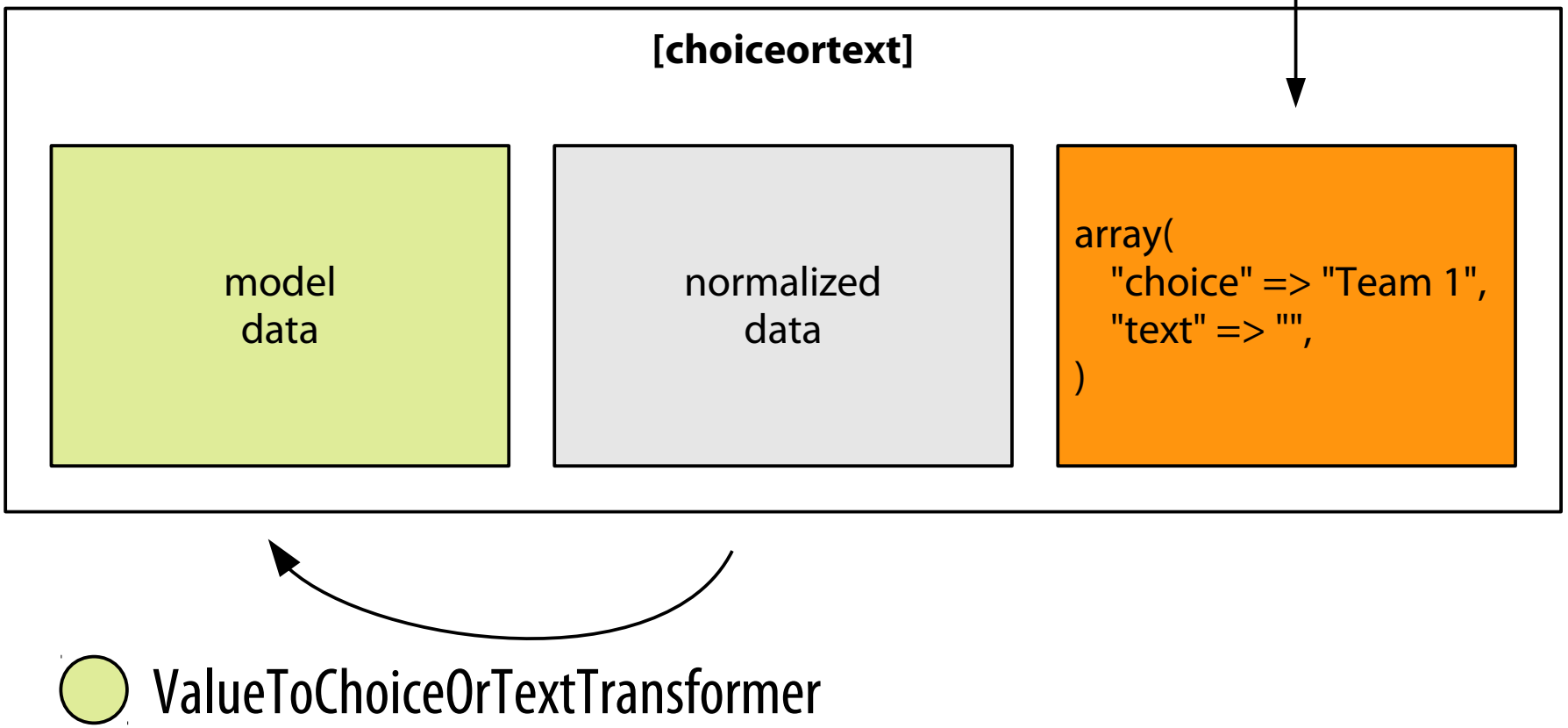
Case 1: Existing Team Selected

```
bind(array(
    'choice' => 'Team 1',
    'text' => '',
));
```



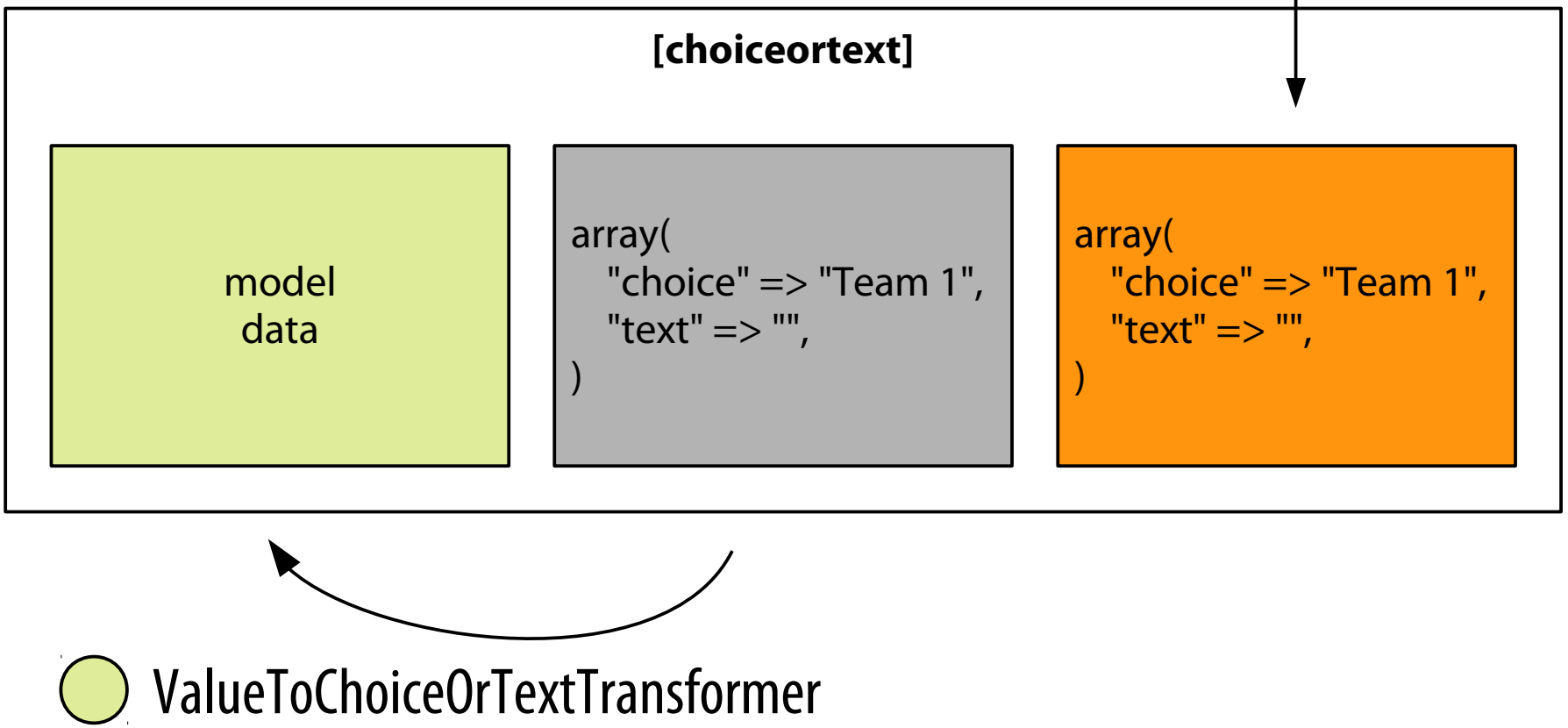
Case 1: Existing Team Selected

```
bind(array(
    'choice' => 'Team 1',
    'text' => '',
));
```



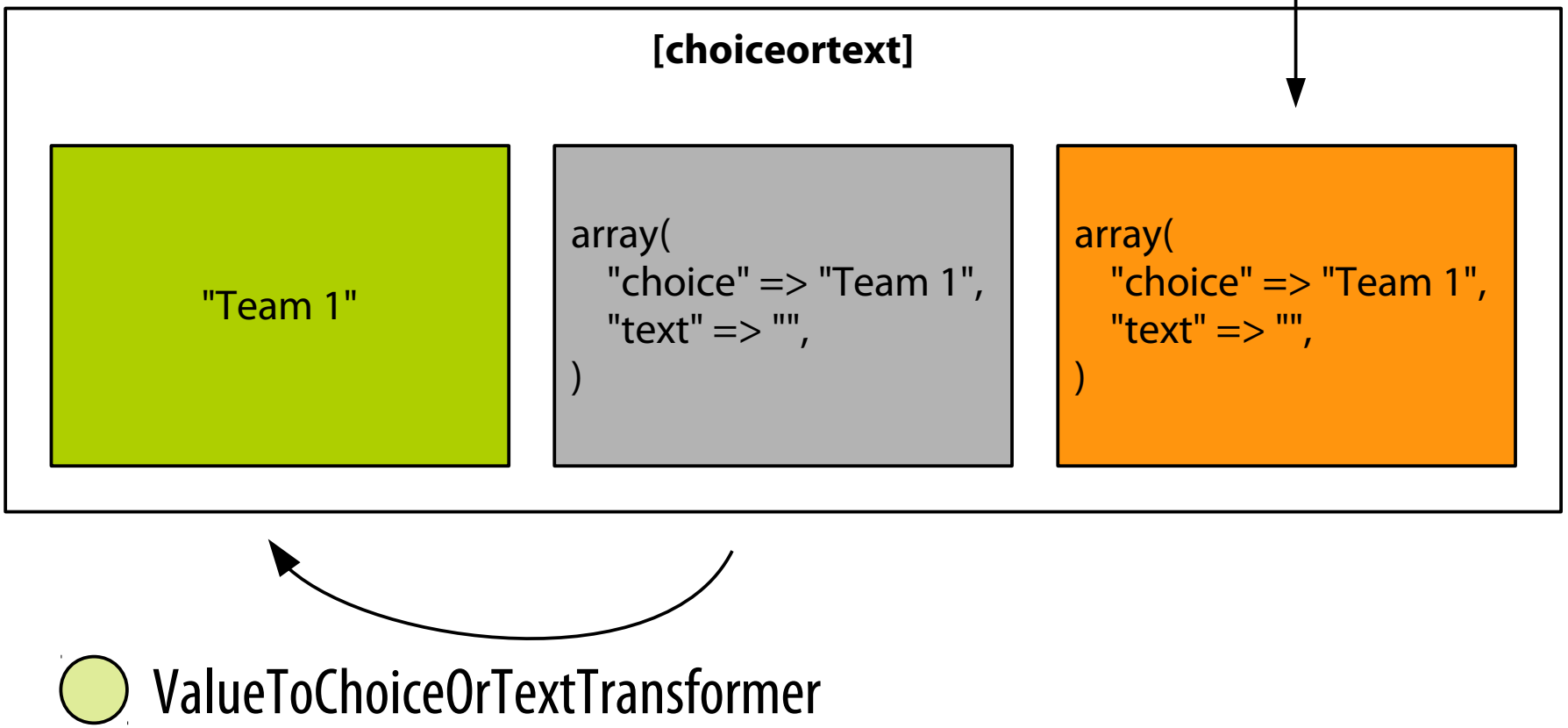
Case 1: Existing Team Selected

```
bind(array(
    'choice' => 'Team 1',
    'text' => '',
));
```



Case 1: Existing Team Selected

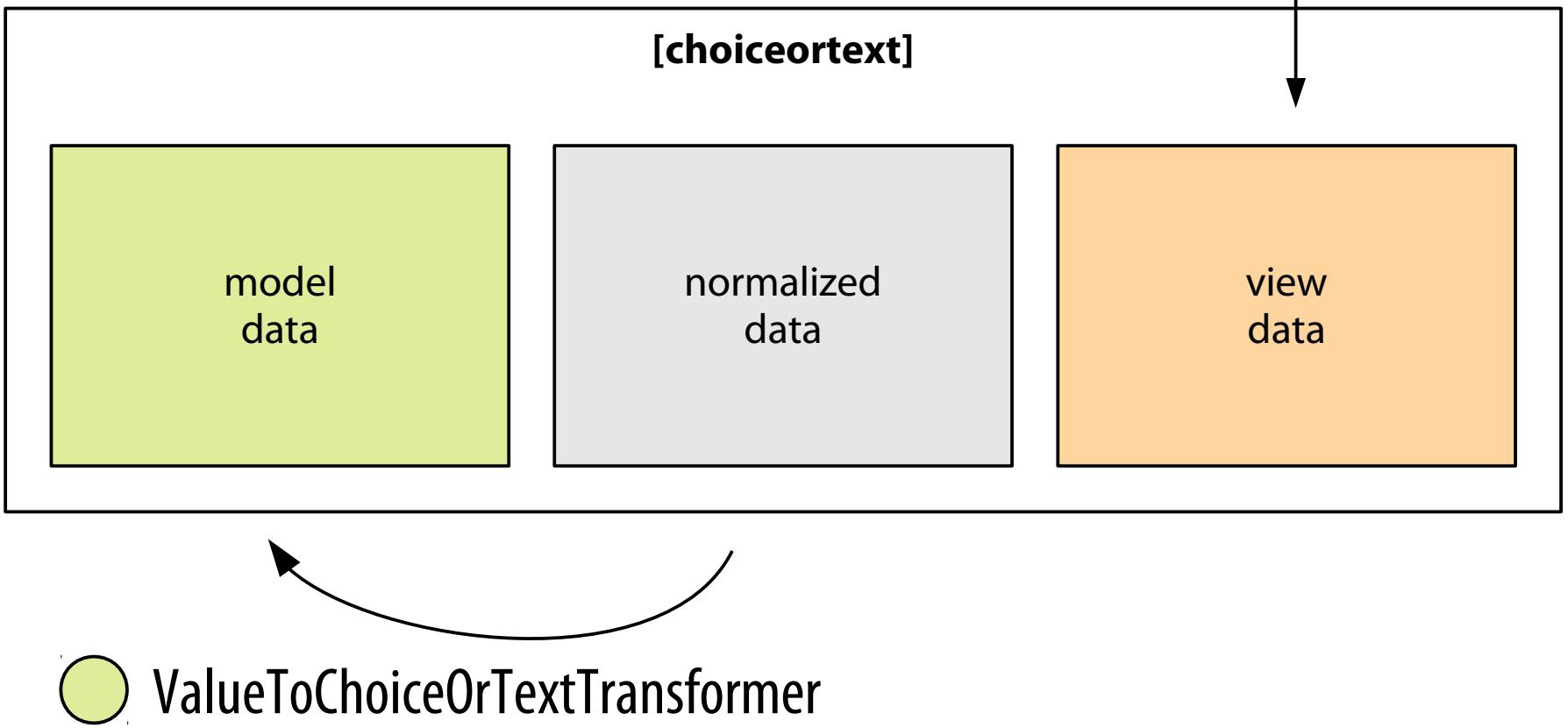
```
bind(array(
    'choice' => 'Team 1',
    'text' => '',
));
```



Case 2: New Team Created

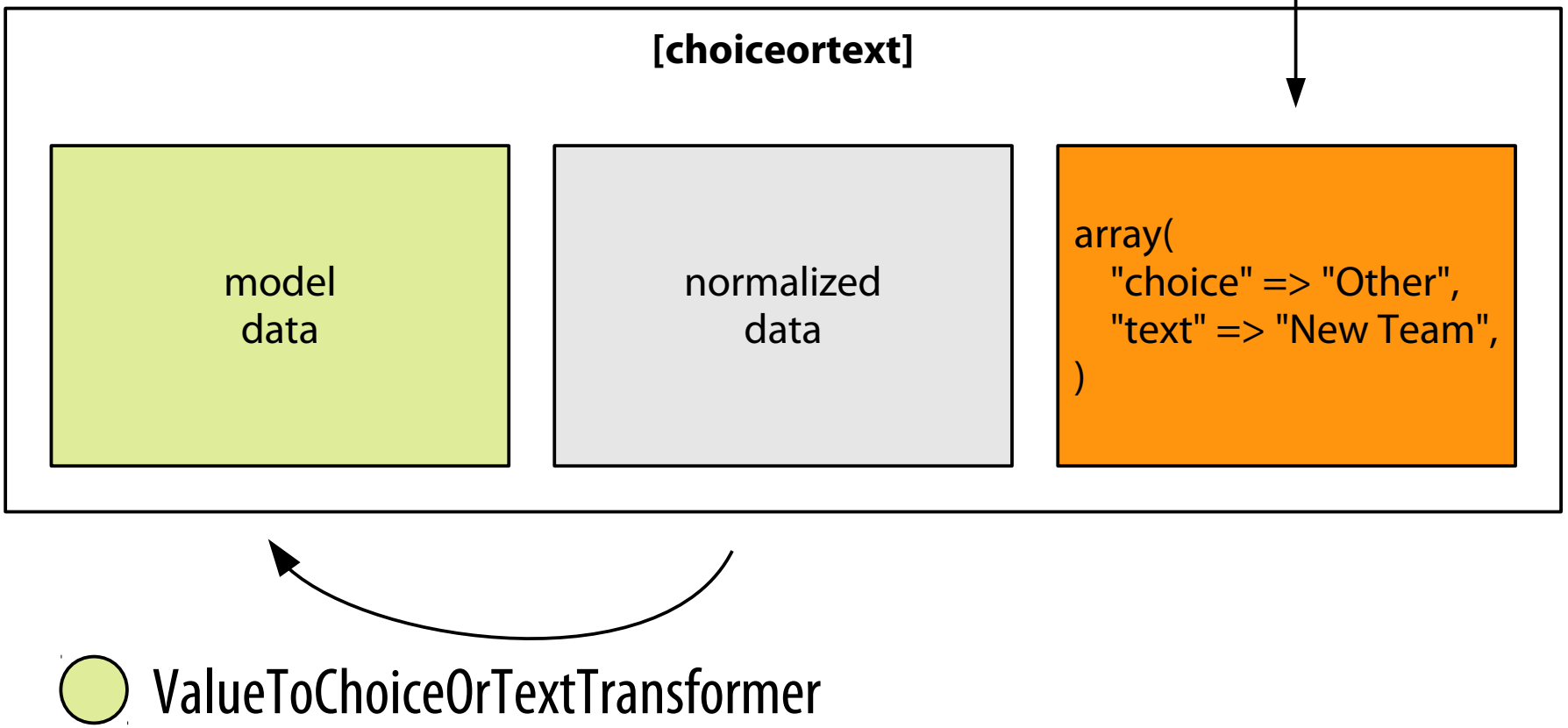
Case 2: New Team Created

```
bind(array(
    'choice' => 'Other',
    'text' => 'New Team',
));
```



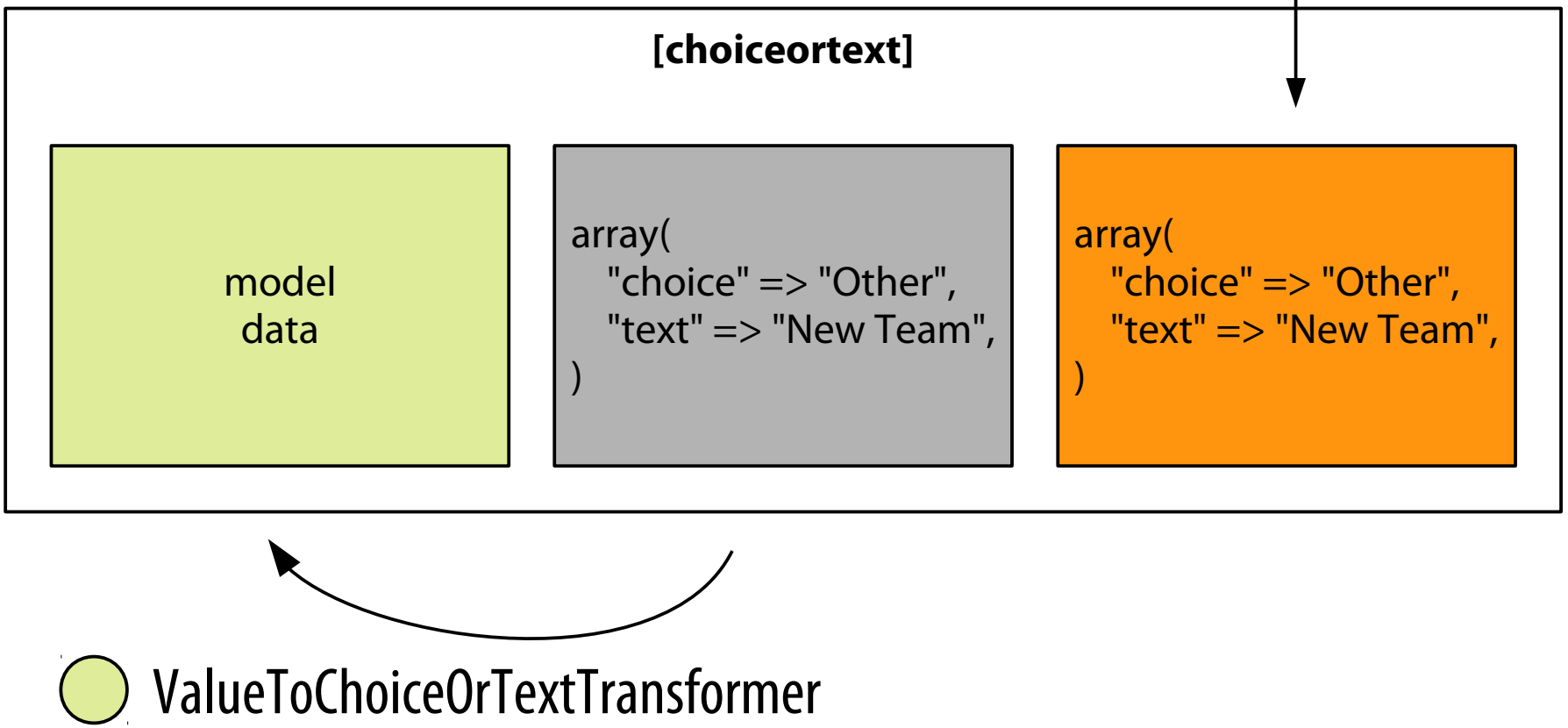
Case 2: New Team Created

```
bind(array(
    'choice' => 'Other',
    'text' => 'New Team',
));
```



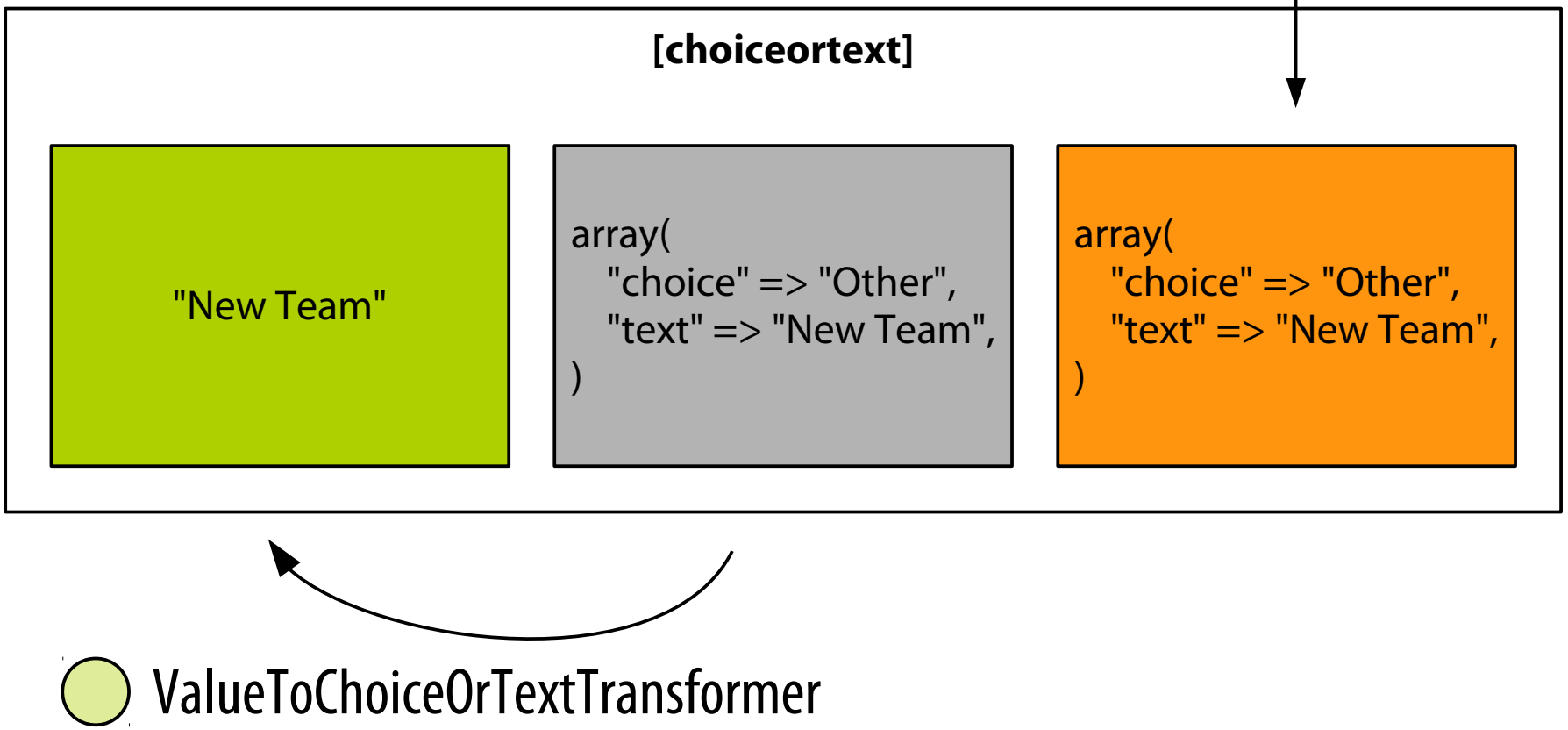
Case 2: New Team Created

```
bind(array(
    'choice' => 'Other',
    'text' => 'New Team',
));
```



Case 2: New Team Created

```
bind(array(
    'choice' => 'Other',
    'text' => 'New Team',
));
```



Tip 

Data transformers should never change the *information* stored in a form, but only the data's *representation*.

To change *information*, use events.

http://www.flickr.com/photos/xverges/4622486563/

Don't

```
array(
    'year' => 2012,
    'month' => 9,
    'day' => 27,
);
```

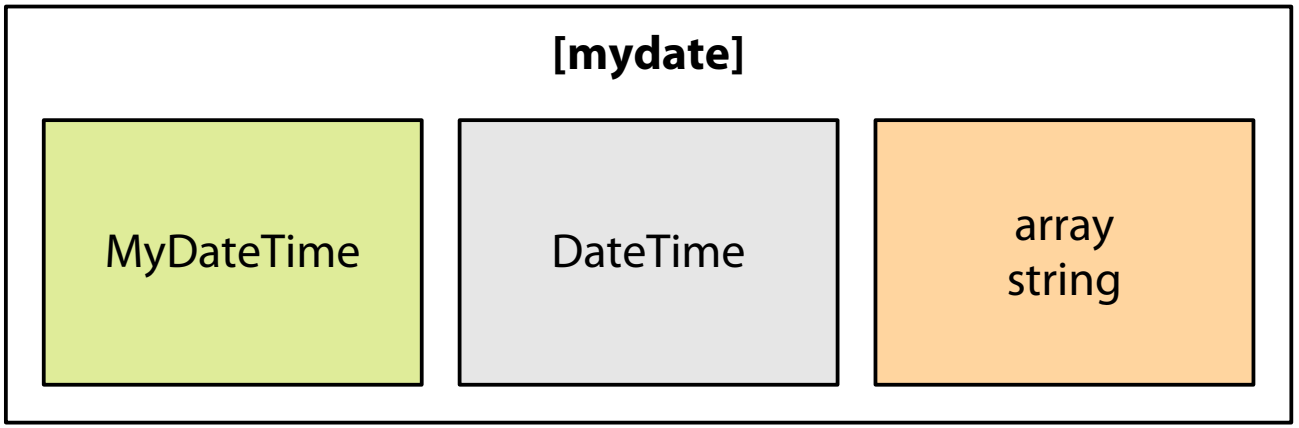
```
array(
    'year' => 2010,
    'month' => 9,
    'day' => 27,
);
```

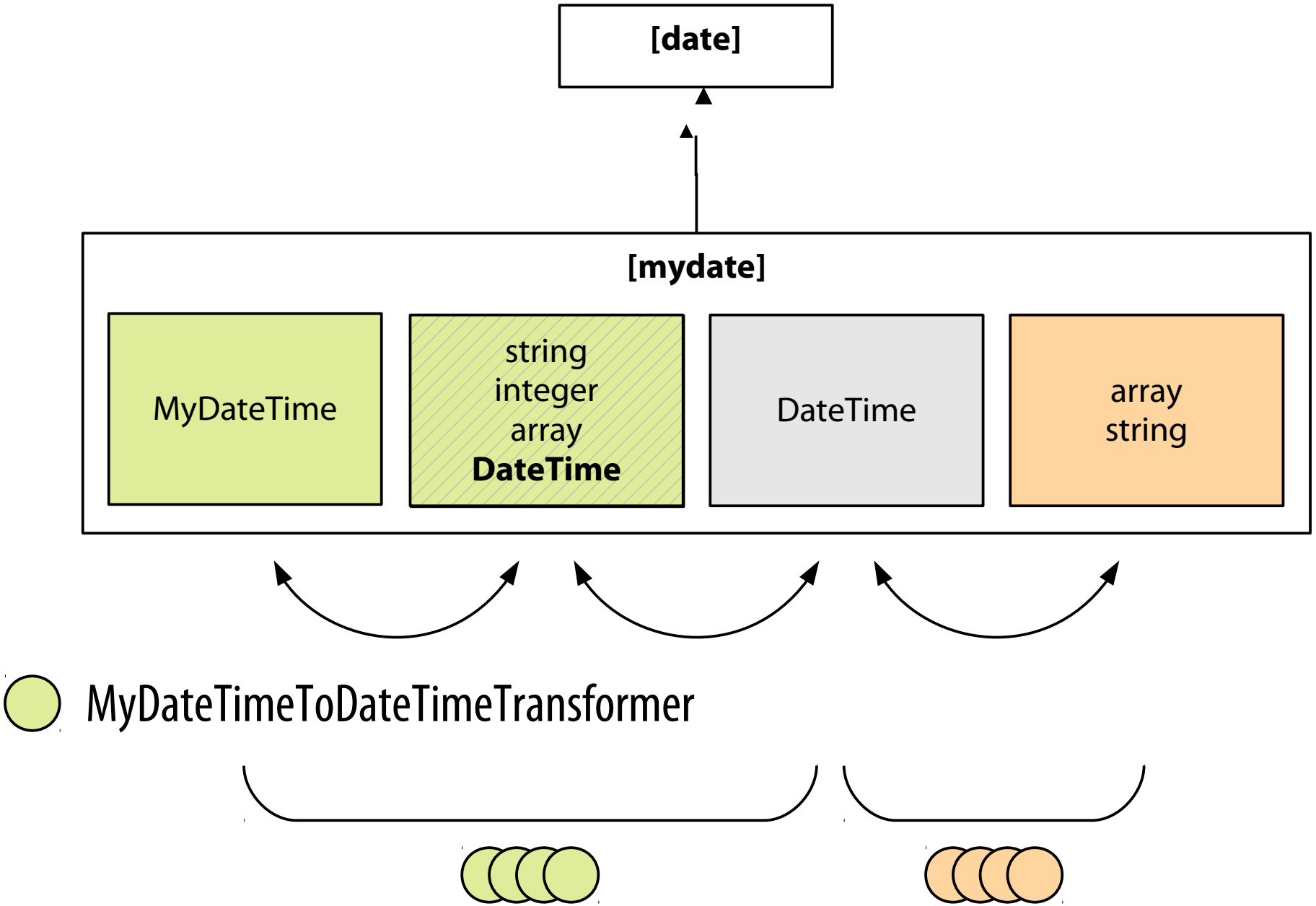
Do

```
array(
    'year' => 2012,
    'month' => 9,
    'day' => 27,
);
```

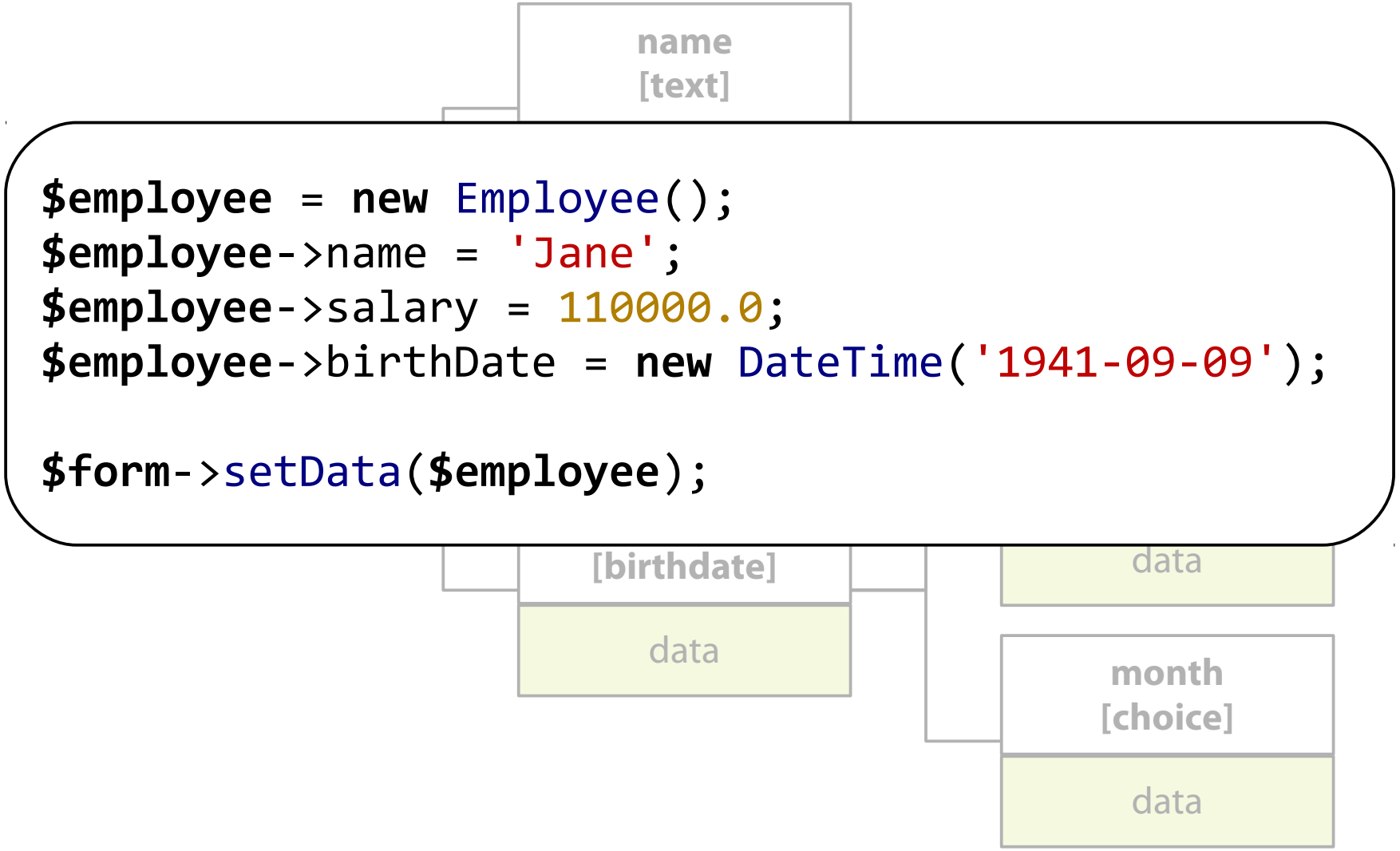
```
DateTime Object
(
    [date] => 2012-09-27 07:52:12
    [timezone_type] => 3
    [timezone] =>
        America/Los_Angeles
)
```

Example: MyDateType



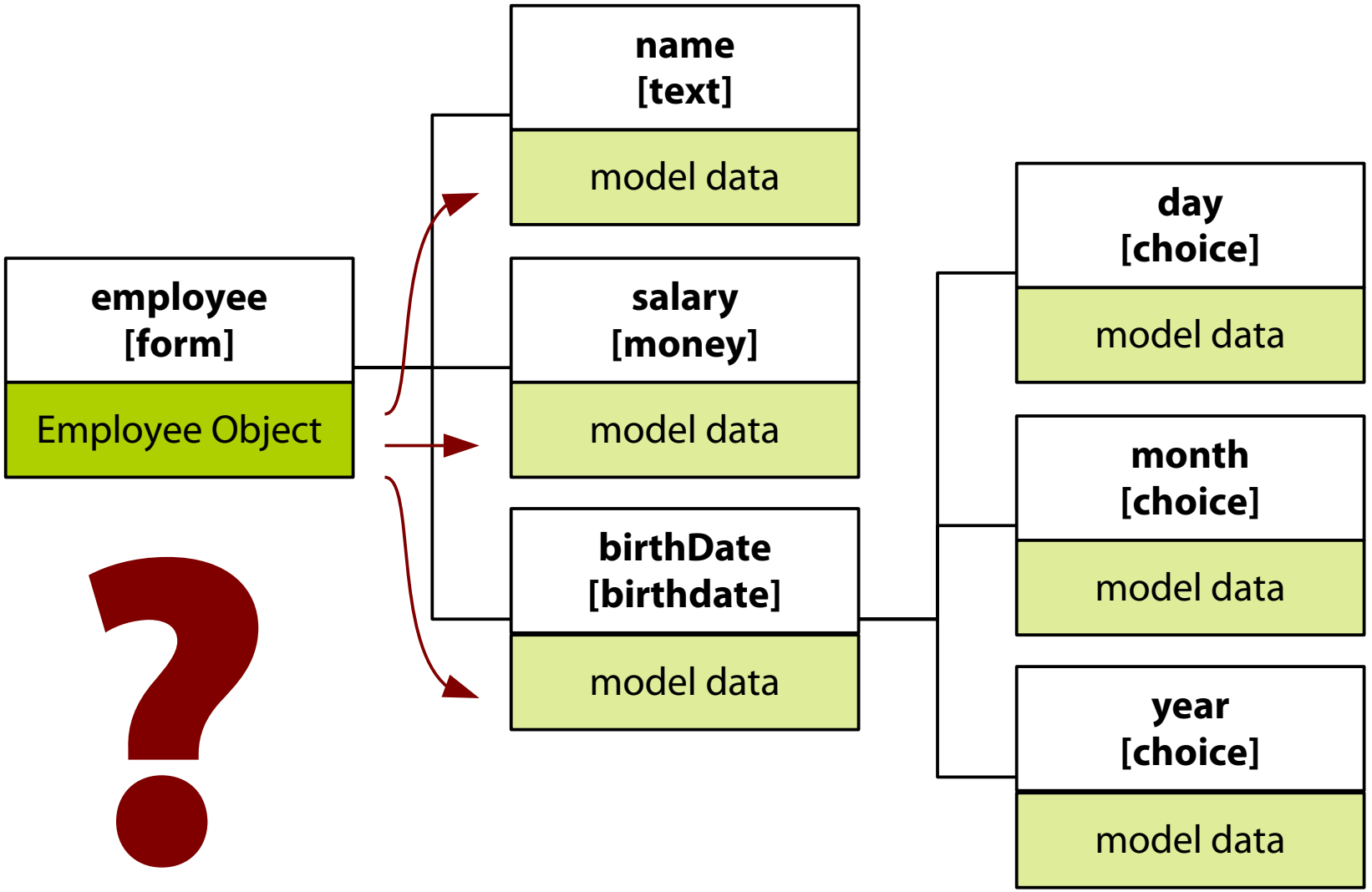

[date]
[mydate]
MyDateTime
string
integer
array
DateTime
DateTime
array
string
MyDateTimeToDateTimeTransformer

Prepopulation

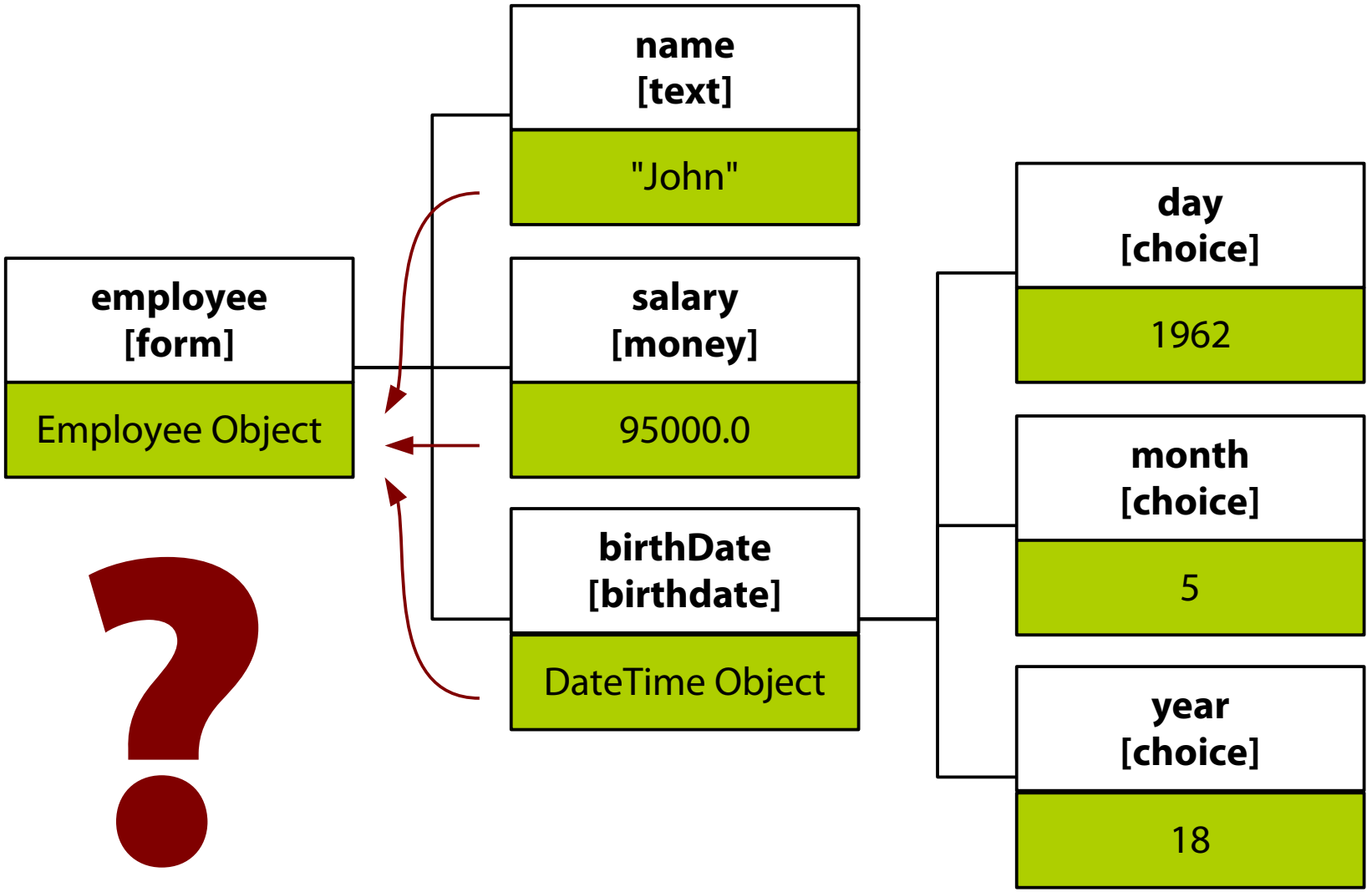



```php
$employee = new Employee();
$employee->name = 'Jane';
$employee->salary = 110000.0;
$employee->birthDate = new DateTime('1941-09-09');

$form->setData($employee);
```

Prepopulation



Binding

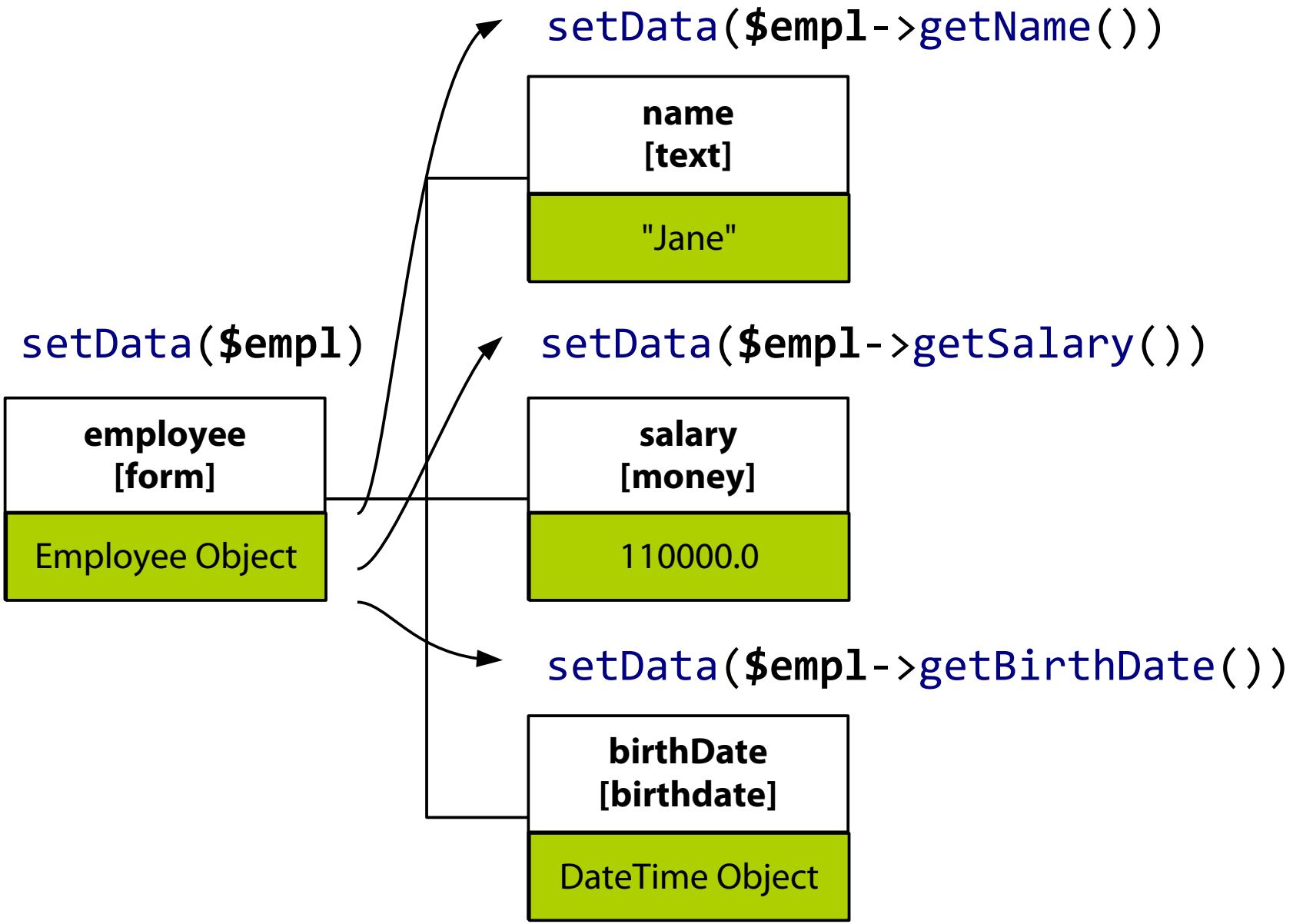

name
[text]
"John"
employee
[form]
Employee Object
salary
[money]
95000.0
birthDate
[birthdate]
DateTime Object
?
day
[choice]
1962
month
[choice]
5
year
[choice]
18

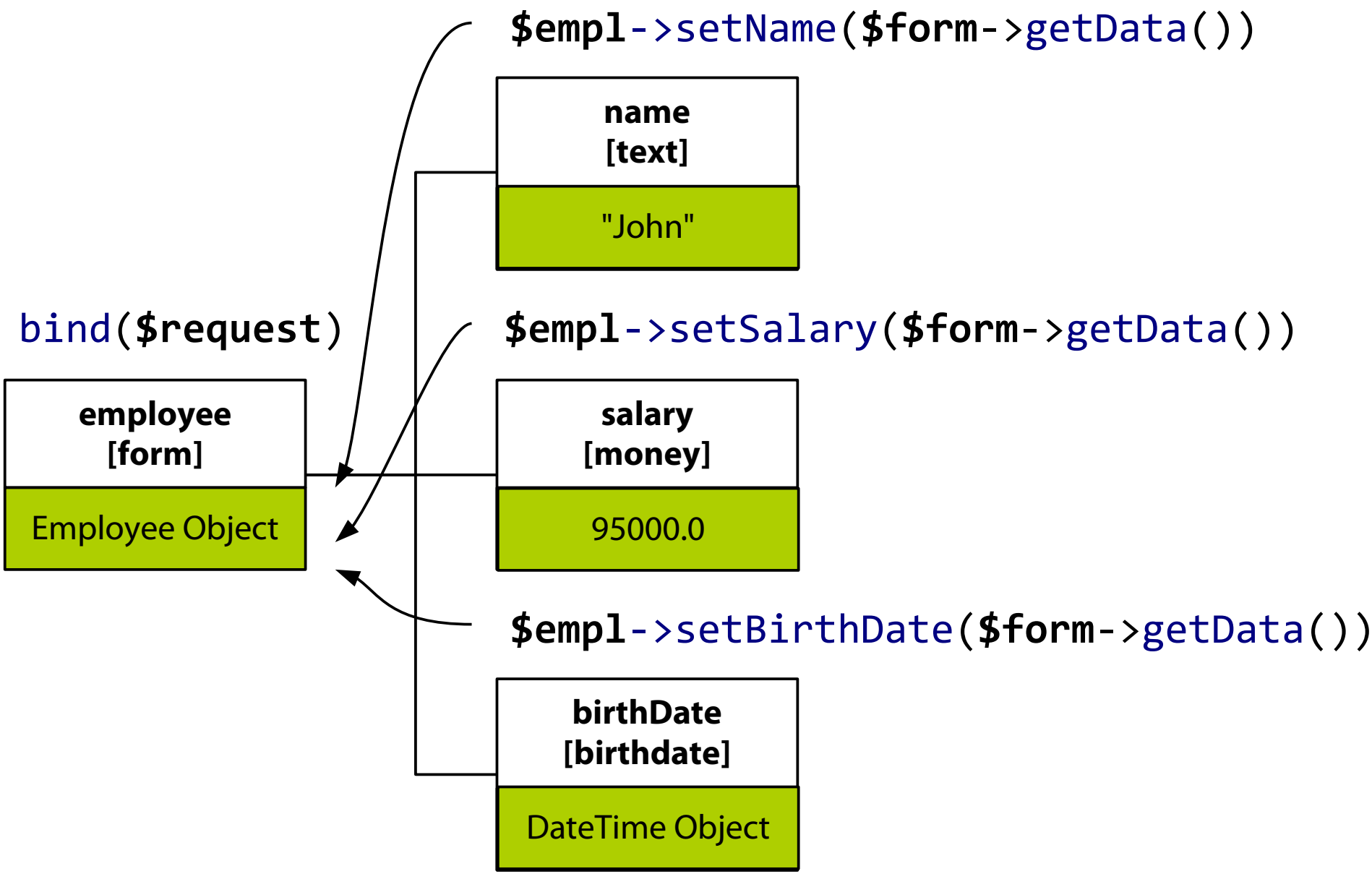
Data Mapping

http://www.flickr.com/photos/wvs/701772889/

Data Mapping

Data Mapping is the exchange of information between a form and its fields.


setData($empl->getName())
name
[text]
"Jane"
setData($empl)
setData($empl->getSalary())
employee
[form]
Employee Object
salary
[money]
110000.0
setData($empl->getBirthDate())
birthDate
[birthdate]
DateTime Object


$empl->setName($form->getData())
name
[text]
"John"
bind($request)
$empl->setSalary($form->getData())
employee
[form]
Employee Object
salary
[money]
95000.0
$empl->setBirthDate($form->getData())
birthDate
[birthdate]
DateTime Object

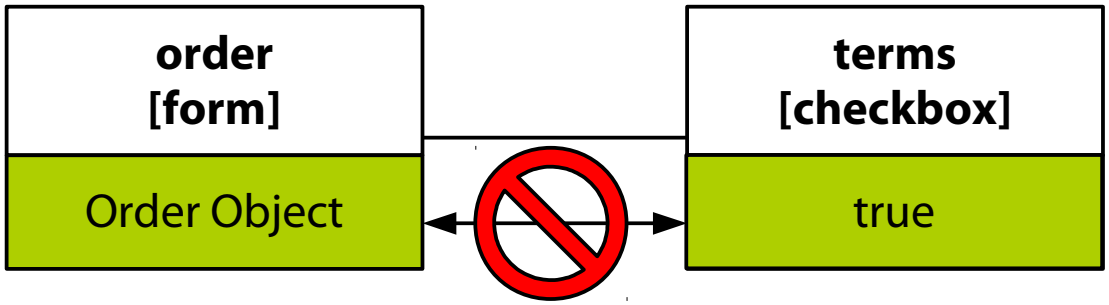
Data Mapping

- Symfony\Component\Form\DataMapperInterface
  - bijective
  - mapFormsToData(array $forms, $data)
  - mapDataToForms($data, array $forms)

Implementations

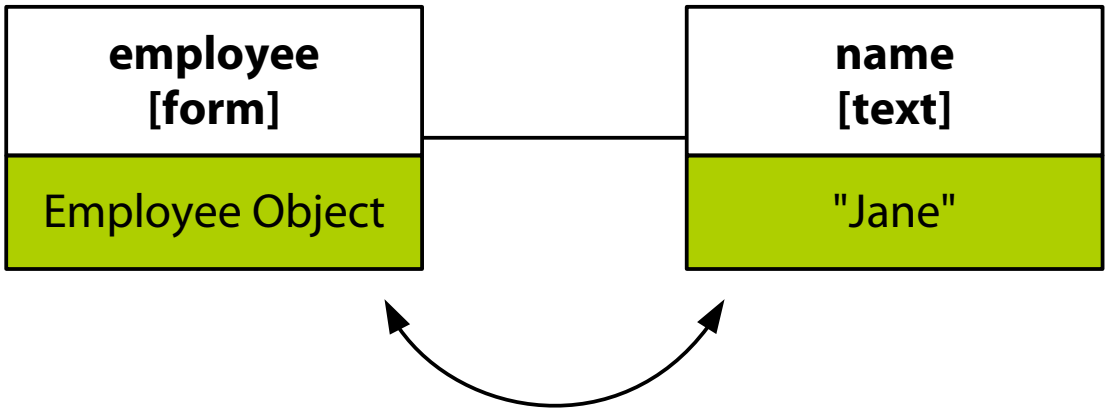
- In Symfony:
  - PropertyPathMapper (getters/setters, arrays)
- Ideas:
  - DomDocumentMapper (nodes of a DomDocument instance)
  - EAVMapper (attributes of EAV instances)

Example: Terms and Conditions




```
$builder->add('terms', 'checkbox', array(
    'mapped' => false,
    'data' => true,
));

$form->get('terms')->setData(true);

$form->get('terms')->getData();
```

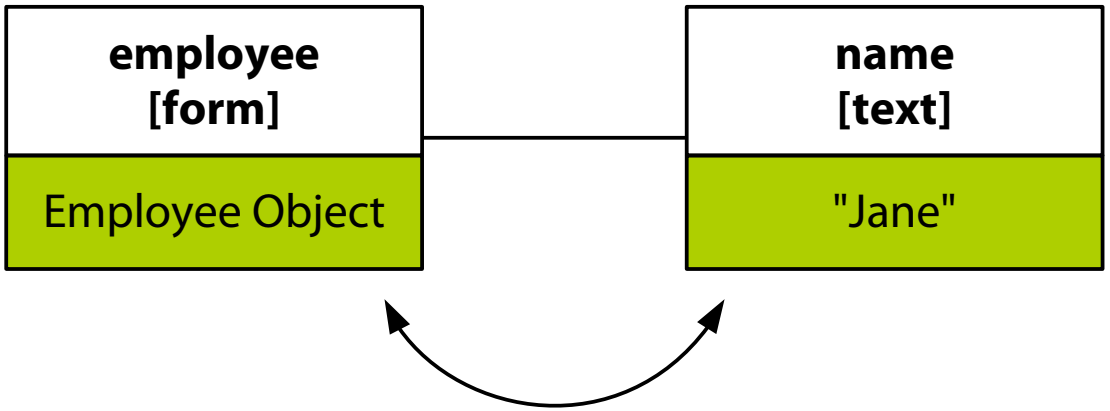
Example: Customized Getters/Setters




```
$form->setData($empl->getFullName())
$empl->setFullName($form->getData())
```

```
$builder->add('name', 'text', array(
    'property_path' => 'fullName',
));
```

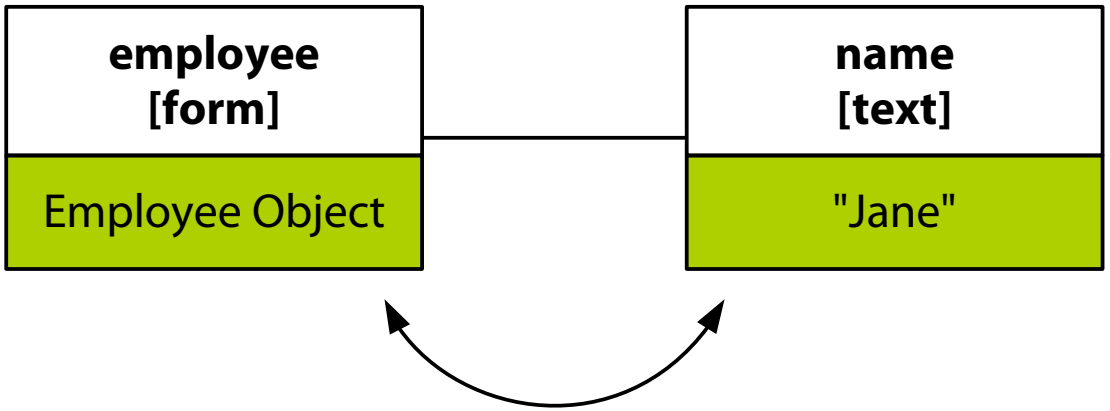
Example: Customized Getters/Setters




```
$form->setData($empl->fullName)
$empl->fullName = $form->getData()
```

```
$builder->add('name', 'text', array(
    'property_path' => 'fullName',
));
```

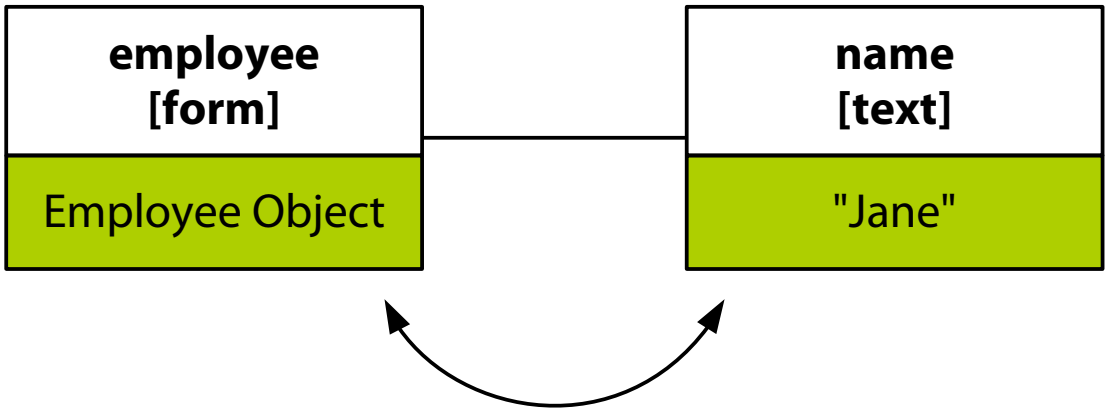
Example: Array Access




```
$form->setData($empl['fullName'])
$empl['fullName'] = $form->getData()
```

```
$builder->add('name', 'text', array(
    'property_path' => '[fullName]',
));
```

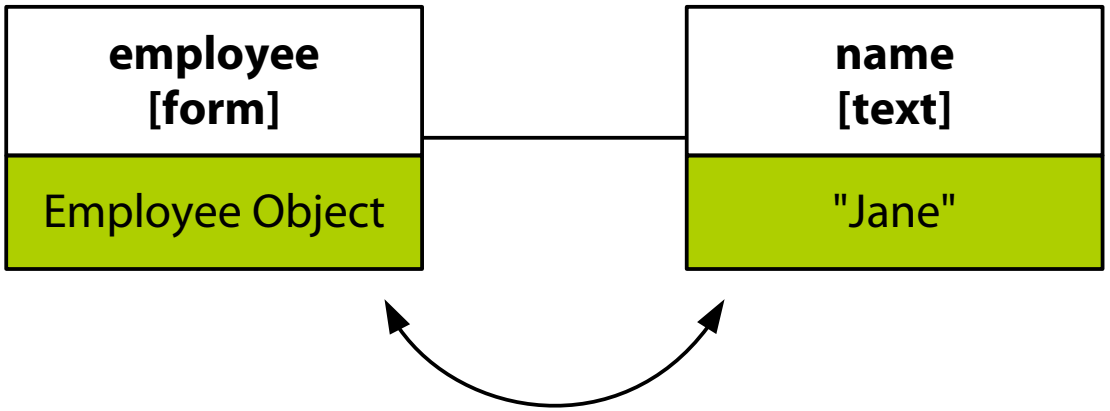
Example: Chained Getters/Setters




```
$form->setData($empl->getPersonalDetails()->getFullName())
$empl->getPersonalDetails()->setFullName($form->getData())
```

```
$builder->add('name', 'text', array(
    'property_path' => 'personalDetails.fullName',
));
```

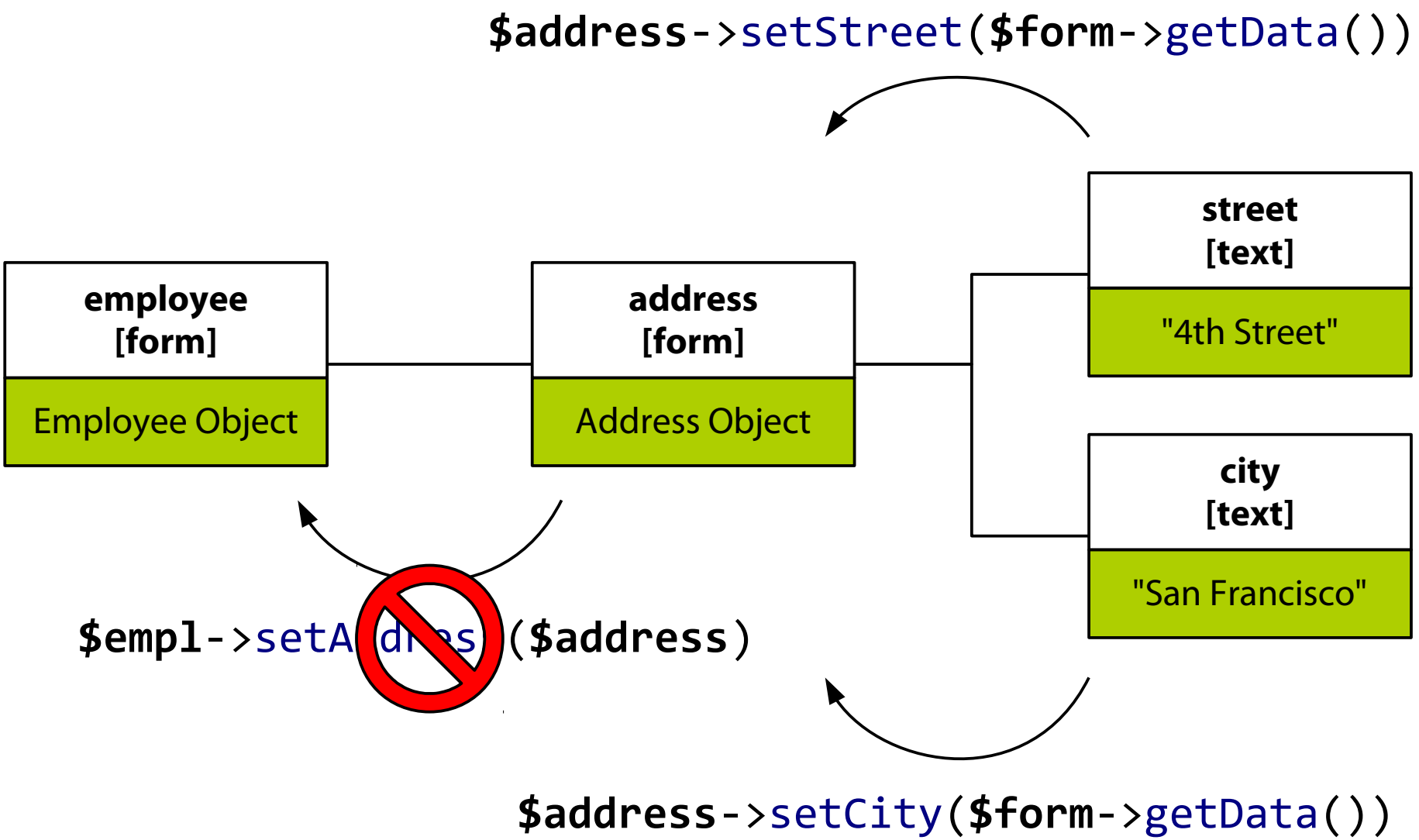
Example: Mixed Chaining




```
$form->setData($empl['personalDetails']->getFullName())
$empl['personalDetails']->setFullName($form->getData())
```

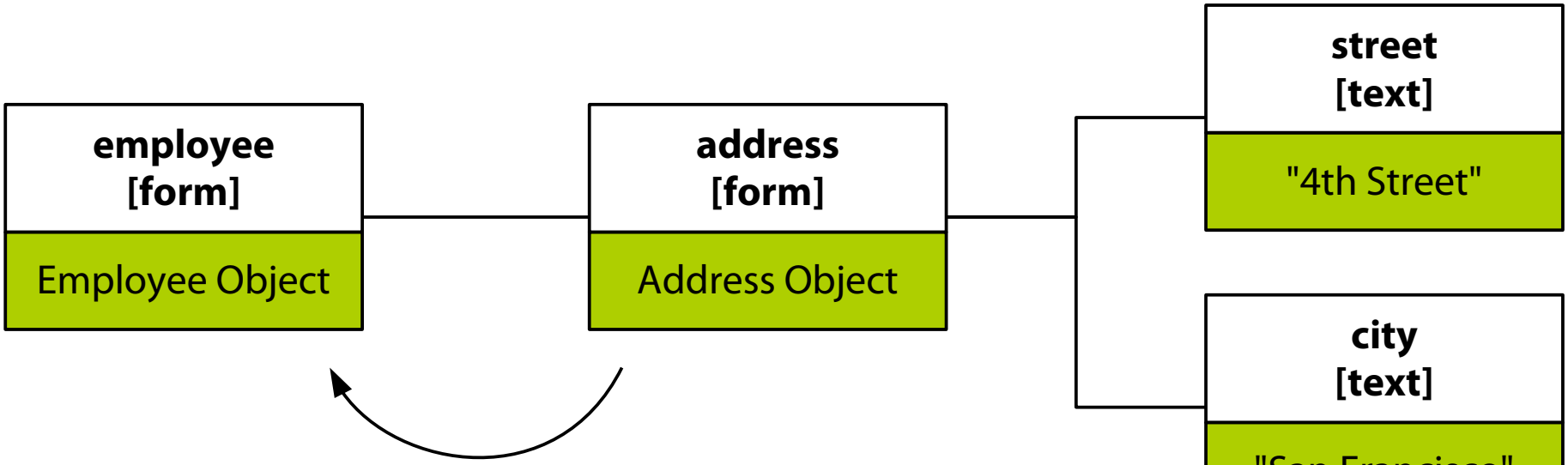
```
$builder->add('name', 'text', array(
    'property_path' => '[personalDetails].fullName',
));
```

Example: Disable By-Reference Handling

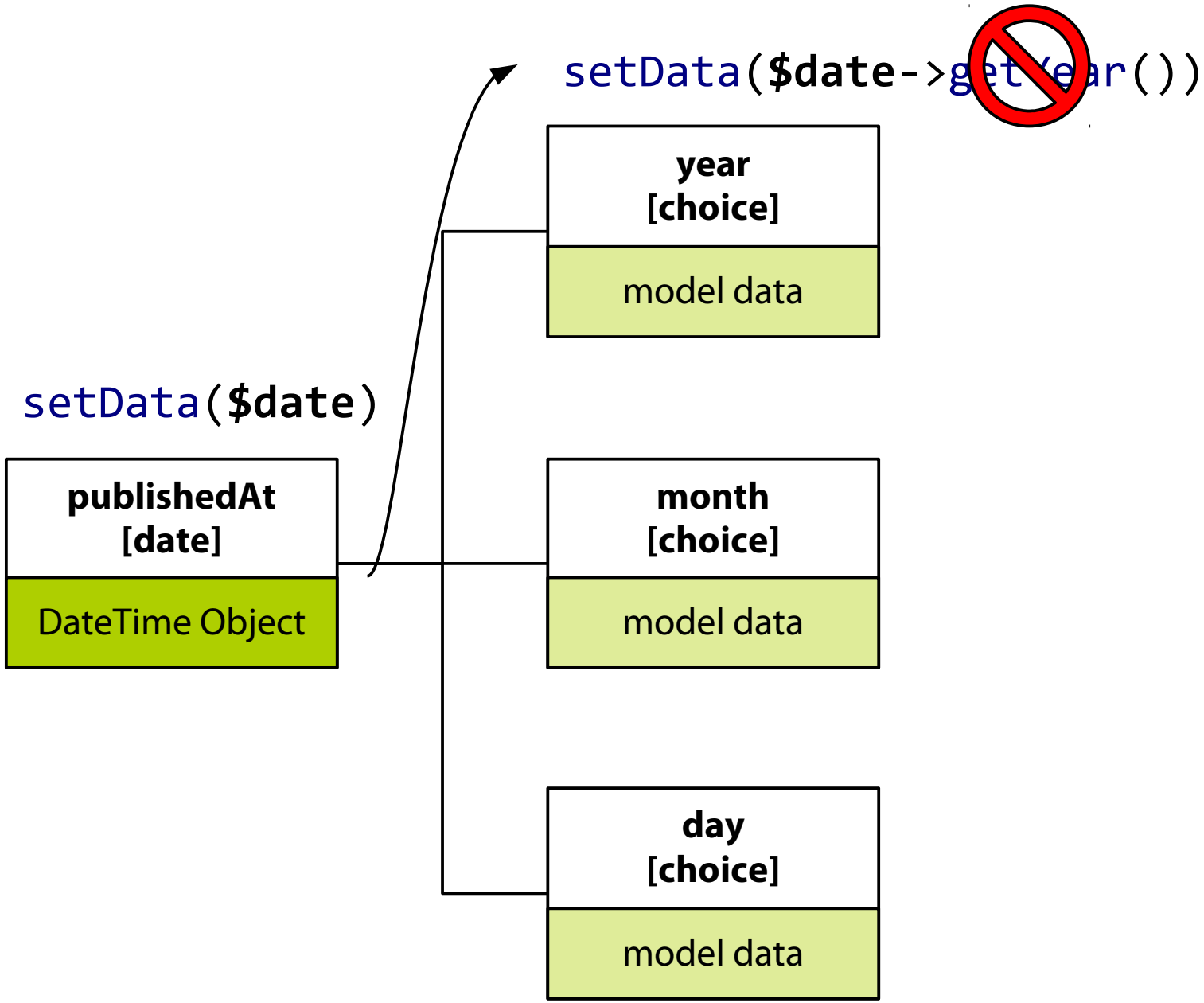


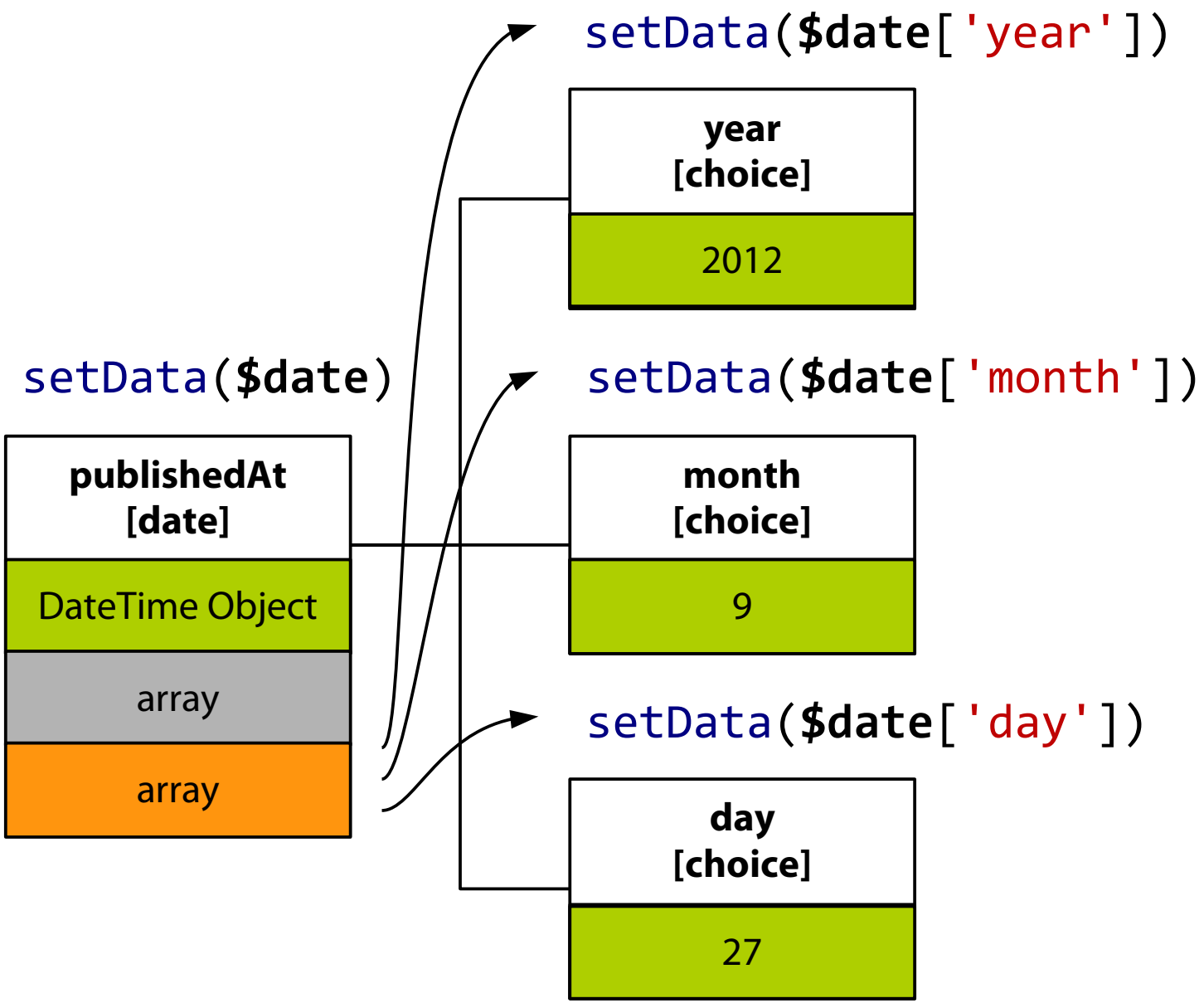
Example: Disable By-Reference Handling

```
$builder->add('address', 'form', array(
    'by_reference' => false,
));
```




```
$empl->setAddress($address)
```


setData($date->getYear())
year
[choice]
model data
setData($date)
publishedAt
[date]
DateTime Object
month
[choice]
model data
day
[choice]
model data


setData($date['year'])
year
[choice]
2012
setData($date)
setData($date['month'])
publishedAt
[date]
DateTime Object
array
array
month
[choice]
9
setData($date['day'])
day
[choice]
27

???

http://www.flickr.com/photos/mom_smackley/4627430439/

Recap

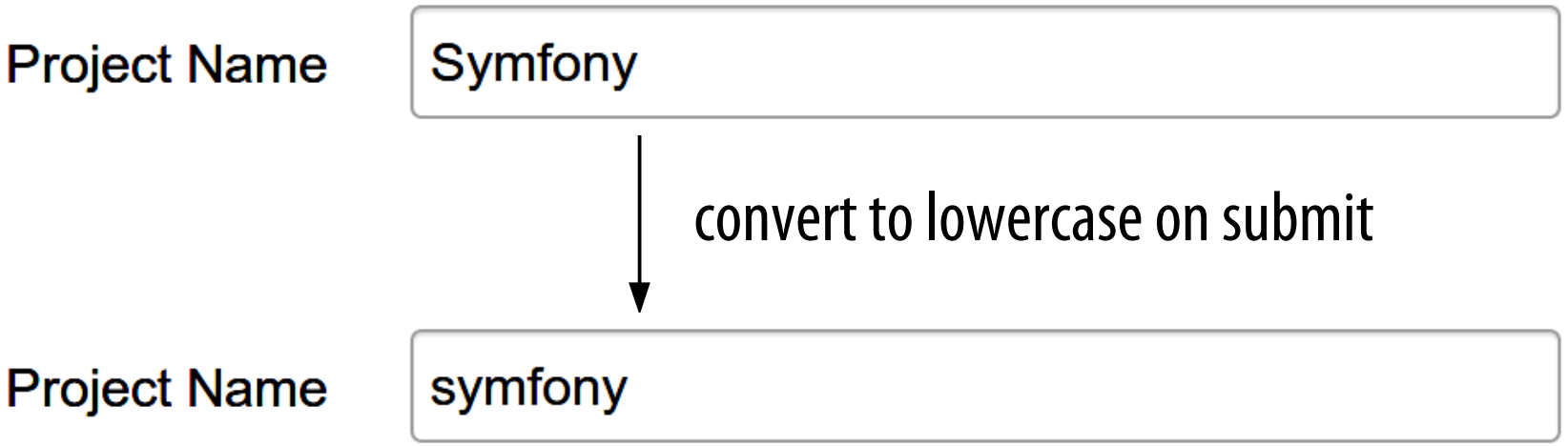
- Data Transformers
  - modification of a data's *representation*
- Data Mappers
  - data exchange between forms and fields

- Missing Piece: Events
  - modification of a data's *content* (filtering, cleaning, ...)
  - dynamic form modification

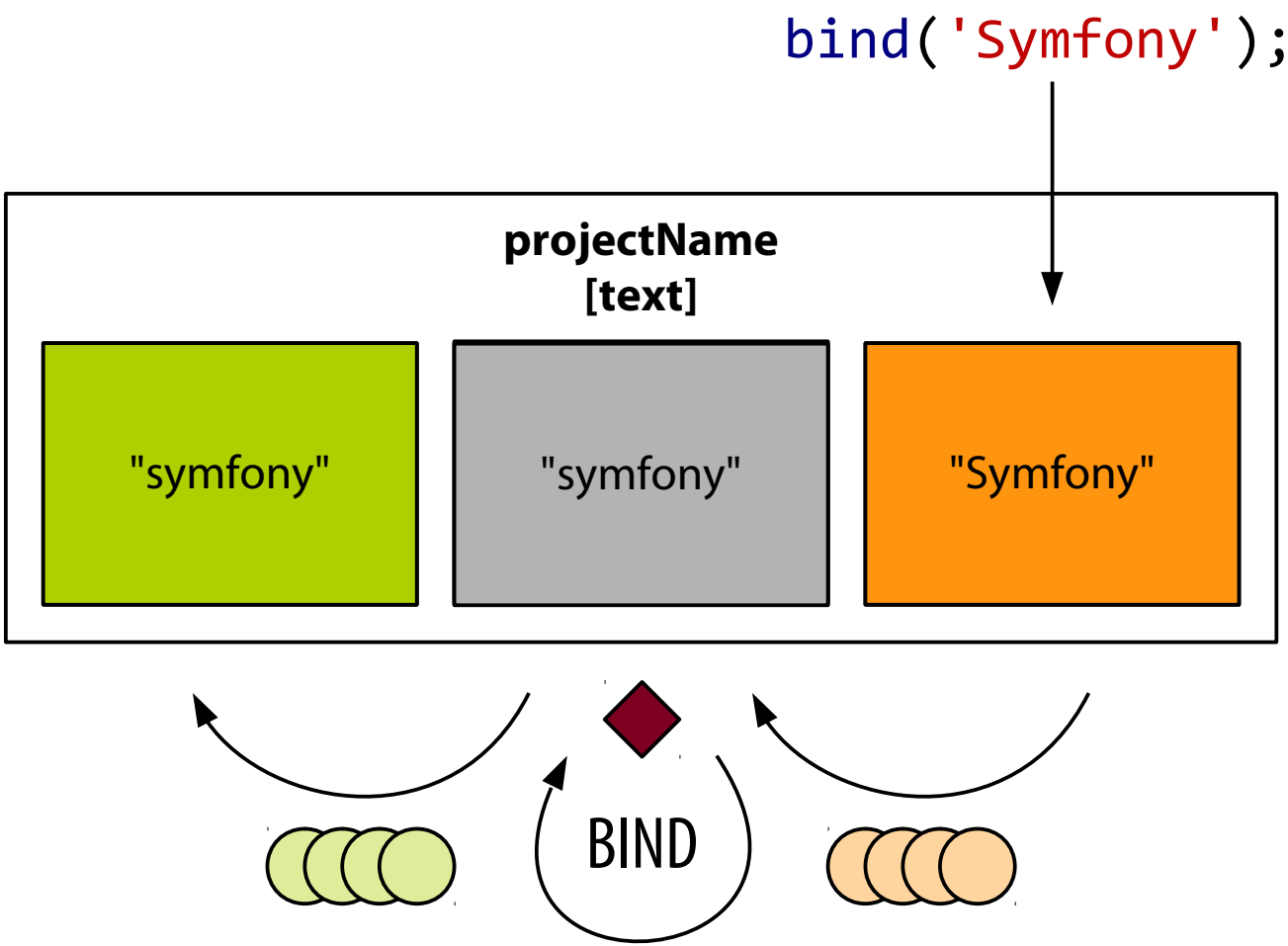
Events

http://www.flickr.com/photos/karamell/4612302824/

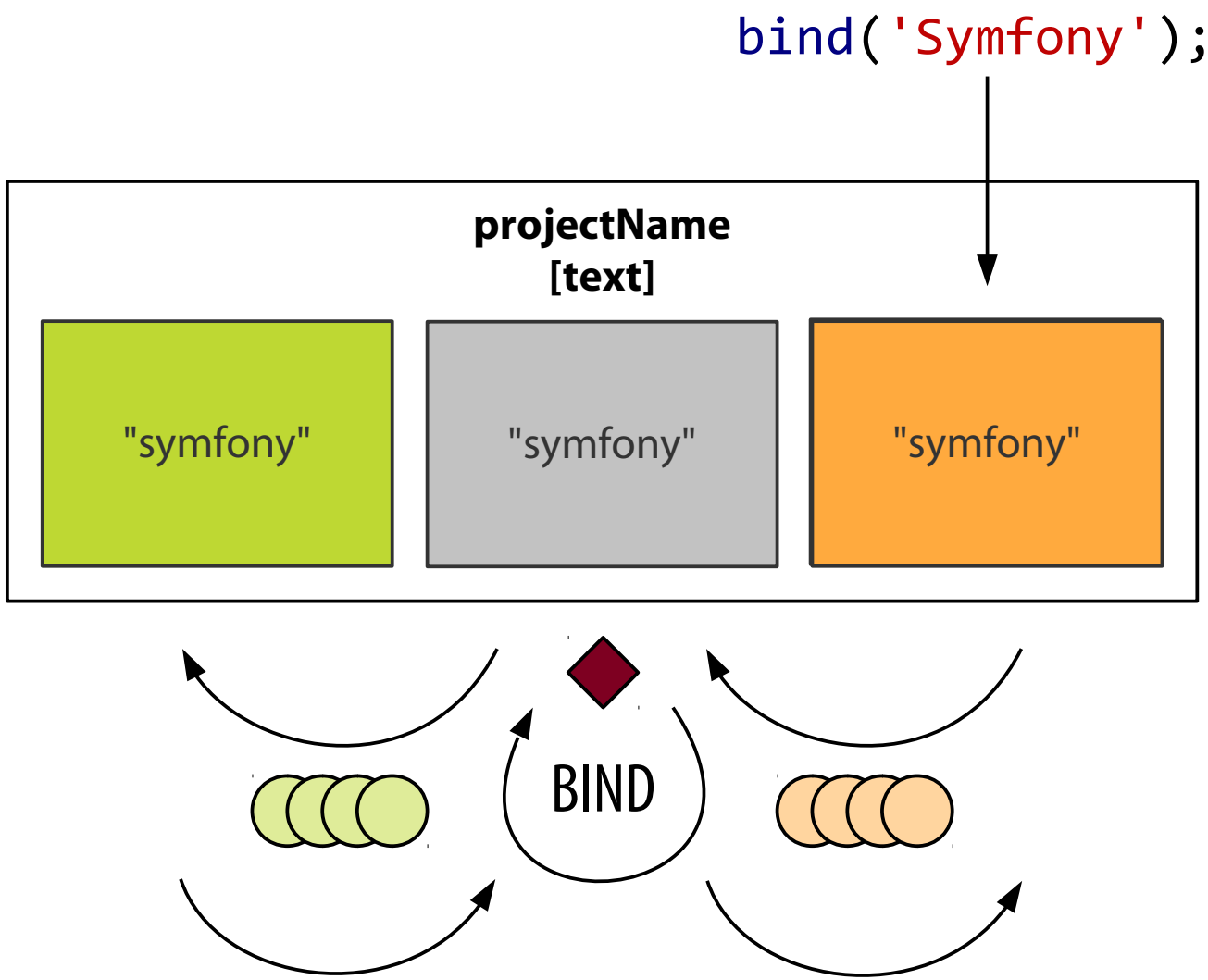
Example: Filtering



Example: Filtering



Example: Filtering



Example: Filtering

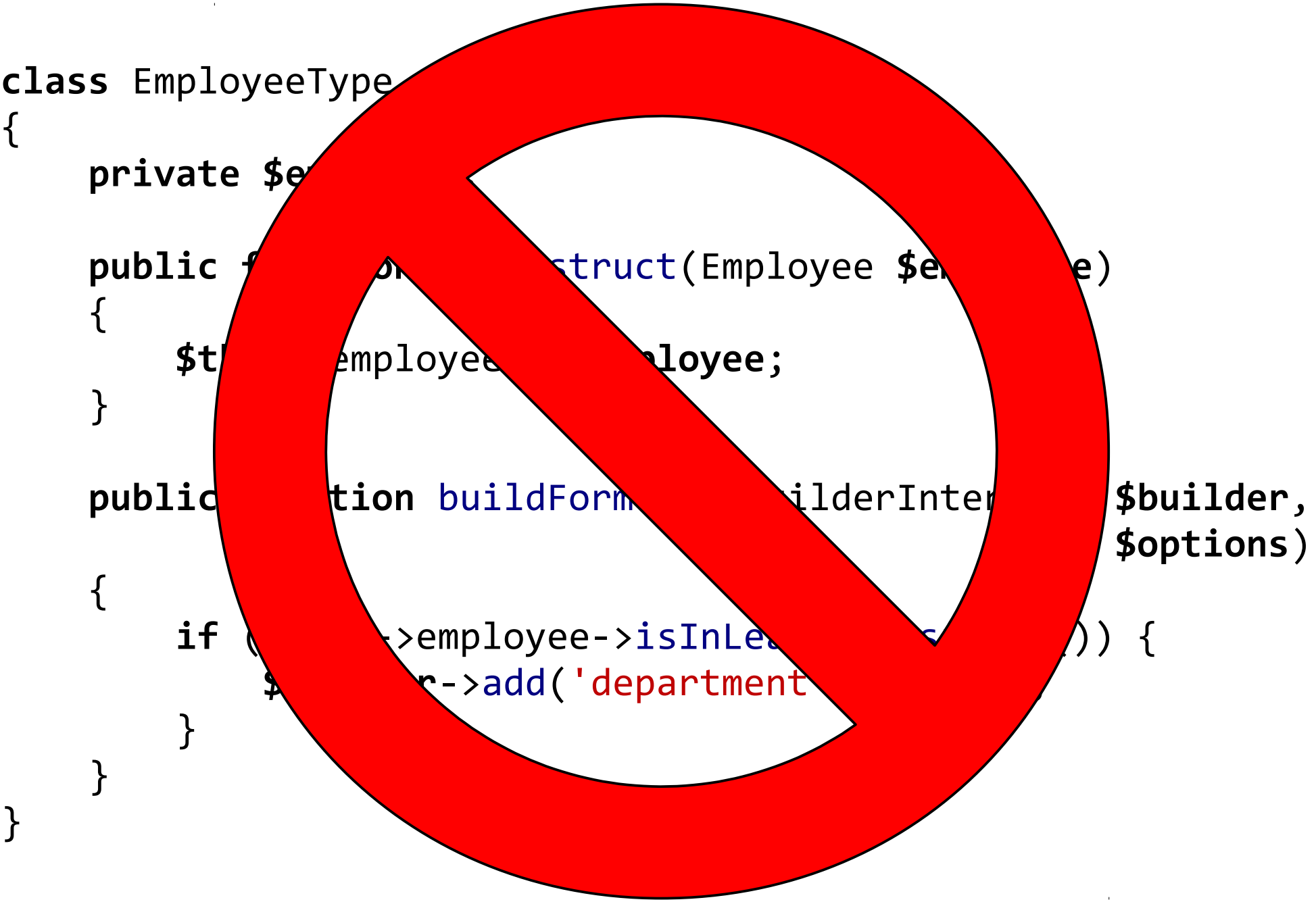
```
$builder->get('projectName')->addEventListener(
    FormEvents::BIND,
    function (FormEvent $event) {
        $event->setData(strtolower($event->getData()));
    }
);
```

Example: Data-Dependent Forms

- Employee in a leading position:

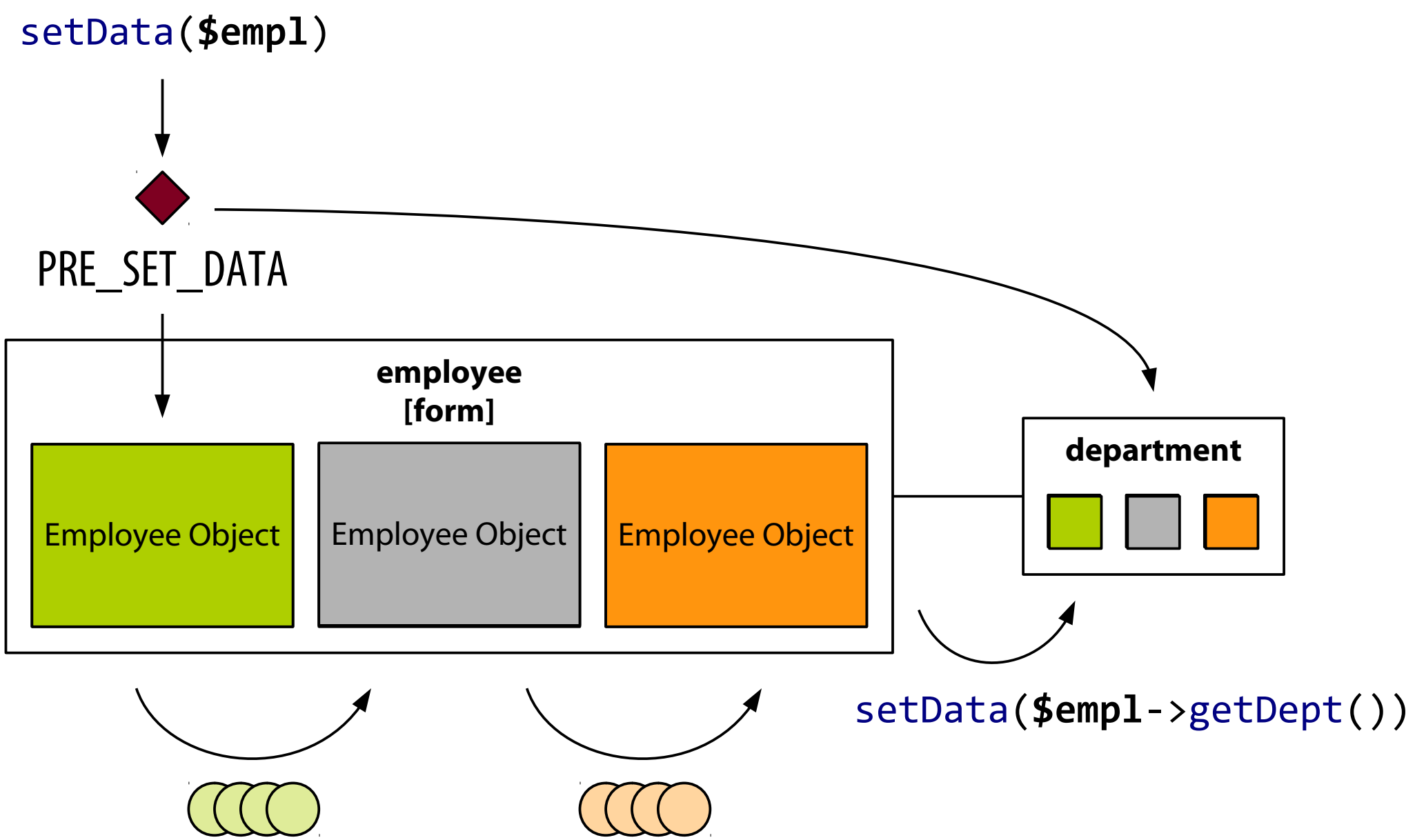
- Employee not in a leading position does not show this field.

```php
class EmployeeType extends AbstractType
{
    private $employee;

    public function __construct(Employee $employee)
    {
        $this->employee = $employee;
    }

    public function buildForm(FormBuilderInterface $builder,
                                                array $options)
    {
        if ($this->employee->isInLeadingPosition()) {
            $builder->add('department', 'text');
        }
    }
}
```


class EmployeeType
{
private $e
public f
$t
public
if
}
}

Form Types vs. Instances

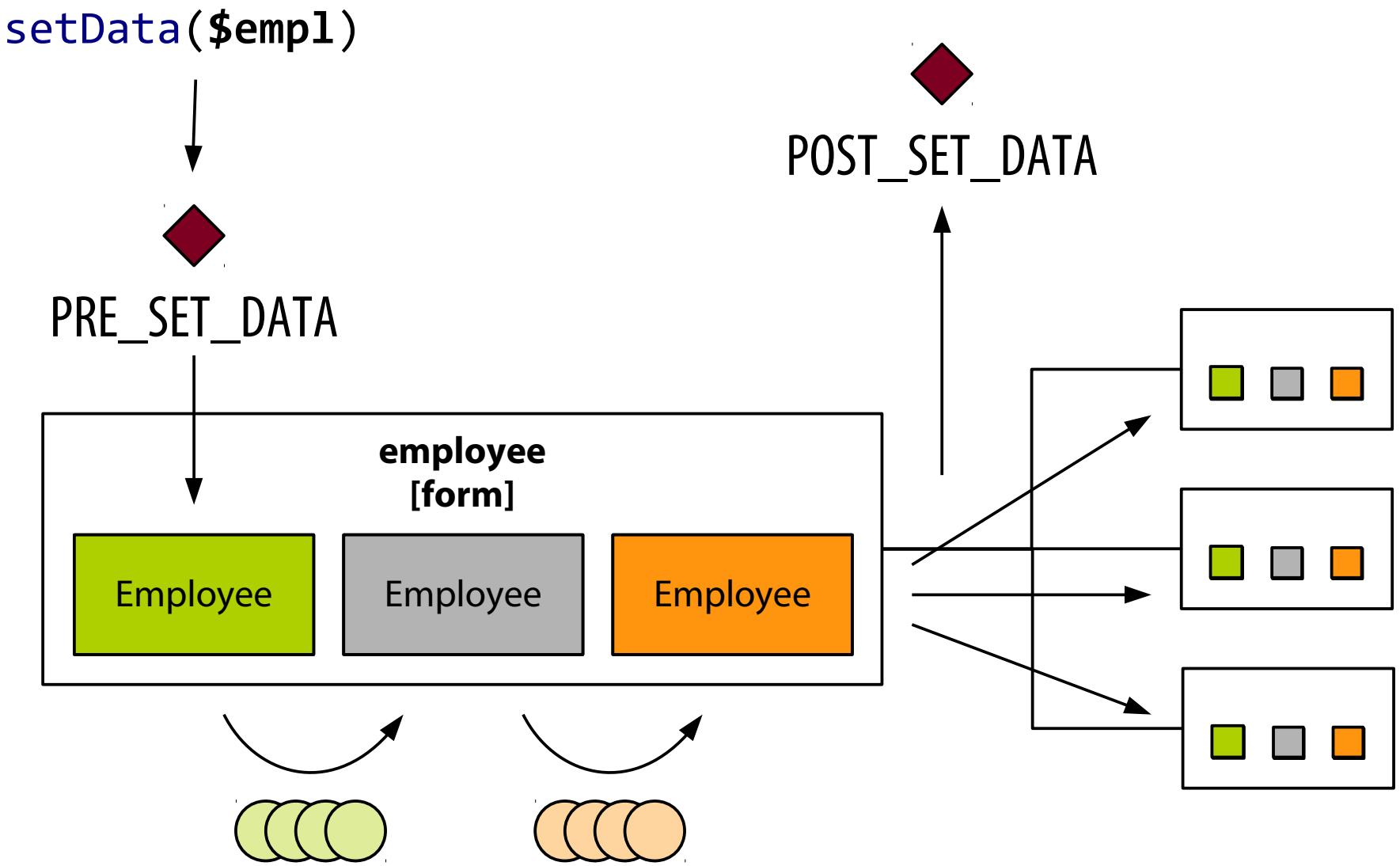
- Form Types
  - exist *once* in your application
  - only dependencies that are the same for every form instance (e.g. EntityManager)
- Form Instances
  - exist multiple times with different data
  - data can (and often will) *change* after construction!!!

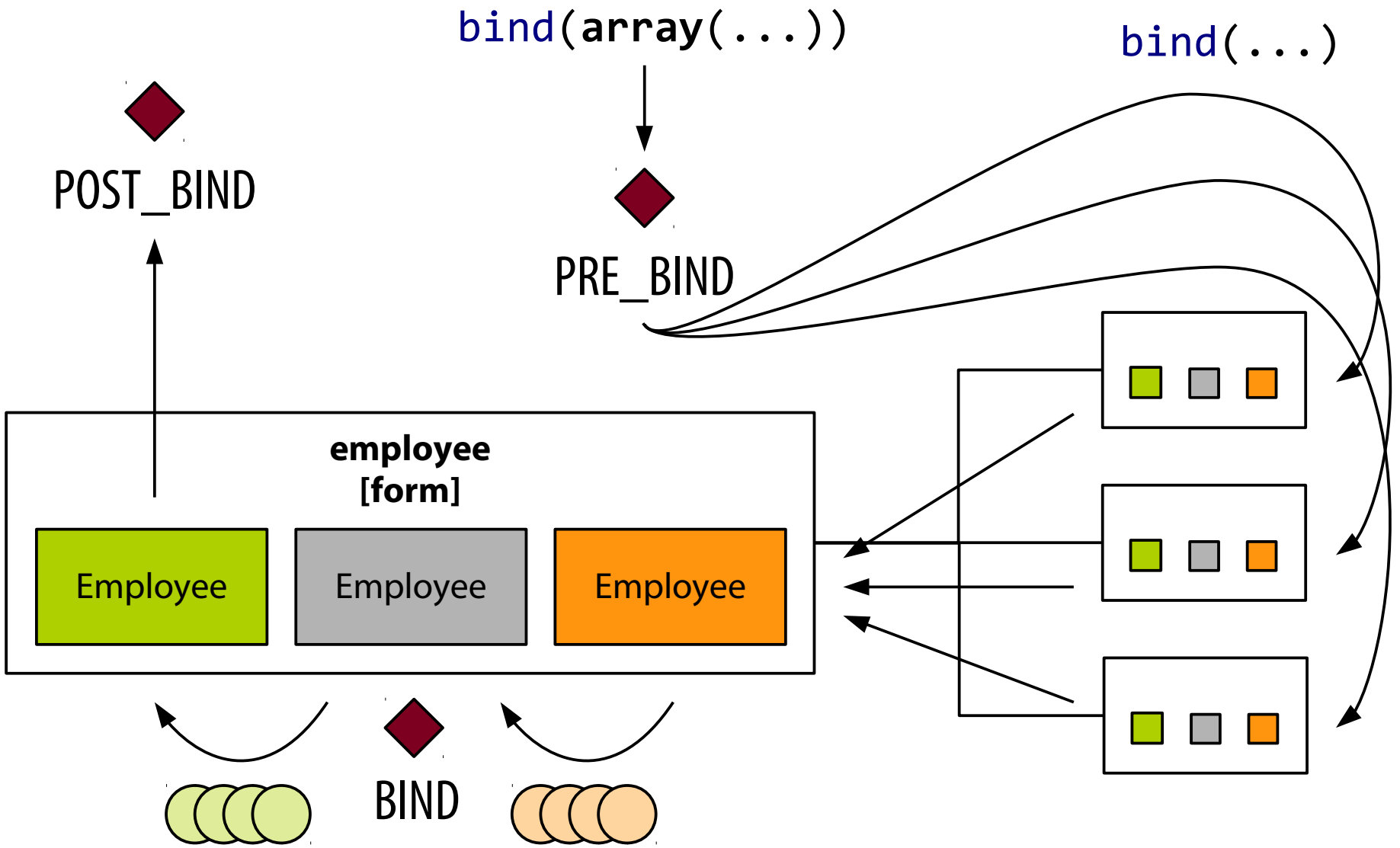

setData($empl)
PRE_SET_DATA
employee
[form]
Employee Object
Employee Object
Employee Object
department
setData($empl->getDept())

```php
public function buildForm(FormBuilderInterface $builder,
                                                    array $options)
{
    $formFactory = $builder->getFormFactory();

    $builder->addEventListener(
        FormEvents::PRE_SET_DATA,
        function (FormEvent $event) use ($formFactory) {
            if ($event->getData()->isInLeadingPosition()) {
                $event->getForm()->add(
                    $formFactory->createNamed('department', 'text');
                );
            }
        }
    );
}
```

Summary

http://www.flickr.com/photos/freefoto/5982549938/


setData($empl)
POST_SET_DATA
PRE_SET_DATA
employee
[form]
Employee
Employee
Employee


bind(array(...))
bind(...)
POST_BIND
PRE_BIND
employee
[form]
Employee
Employee
Employee
BIND


Thank you!
https://joind.in/talk/view/7217
Bernhard Schussek
@webmozart