

# Zend\_Microformat\_Xfn - Pádraic Brady

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

## Zend Framework: Zend\_Microformat\_Xfn Component Proposal

<b>Proposed Component Name</b>	Zend_Microformat_Xfn
<b>Developer Notes</b>	<a href="http://framework.zend.com/wiki/display/ZFDEV/Zend_Microformat_Xfn">http://framework.zend.com/wiki/display/ZFDEV/Zend_Microformat_Xfn</a>
<b>Proposers</b>	Pádraic Brady Matthew Weier O'Phinney (Zend Liaison)
<b>Revision</b>	1.0 - Awaiting Community Comment (wiki revision: 6)

## Table of Contents

1. Overview
2. References
3. Component Requirements, Constraints, and Acceptance Criteria
4. Dependencies on Other Framework Components
5. Theory of Operation
6. Milestones / Tasks
7. Class Index
8. Use Cases
9. Class Skeletons

## 1. Overview

The XHTML Friends Network (XFN) Microformat was designed to enable users classify their relationships with other individuals and entities they have linked to. This is the sort of data which can be used in a social networking application to promote individuals you have linked to into categories based on your personal relationships.

A simple example is that of two individuals, Bob and Ann. Bob views Ann as a remote friend, and creates a link as follows:

```
<a href="http://www.example.com/ann" rel="friend met">Ann</a>
```

The rel attributes indicate personal relationships. In this case Bob views Ann as a friend he has met in person. Ann might have slightly different ideas:

```
<a href="http://www.example.com/bob" rel="friend crush met">Ann</a>
```

This time Ann indicates that only does she consider Bob as a friend she has met in person, but that she also has a crush on him!

The simplicity of the format enables social application platforms to potentially generate a graph of all relationships any individual has, and their inverse linkage - something Google has recently announced... But also, it's a simple means of collecting any individuals contacts, friends and acquaintances which is already usable on some platforms for allowing a new member to invite and import their networks from other social apps.

## 2. References

- [XFN Introduction And Examples](#)
- [Microformats.org XFN Wiki](#)
- [About Microformats](#)

## 3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** parse XFN data from any given URL.
- This component **will** parse XFN data from any given HTML/XHTML source.
- This component **will** generate XFN enabled hyperlinks for embedding in HTML.
- This component **will** present and accept data using an iterable and countable OO API.

## 4. Dependencies on Other Framework Components

- `Zend_Microformat`
- `Zend_Microformat_Exception`

## 5. Theory of Operation

The operation of this component is covered in the related `Zend_Microformat` proposal. In short, when supplied with a URL or HTML source, this component will parse out XFN data into a results object for use. It will also generate similar embeddable output.

Although this component can be used directly for specific purposes, or for implementing custom XFN forms, the majority of use cases work perfectly fine using the API supplied with `Zend_Microformat`.

The use cases below present this component's specific usage, a `Zend_Microformat` preferred alternative, and a generation example.

As with all Microformats, this component is subject to legal usage as defined in the relevant XFN 1.1 specification. Where an attempt to add data to an XFN Entity breach the specification, the error will raise an Exception.

## 6. Milestones / Tasks

- Milestone 1: Assemble some initial use cases to explore desired API
- Milestone 2: Commence proof-of-concept coding (will discard after)
- Milestone 3: Pending review, complete initial development using TDD
- Milestone 4: Complete acceptance testing, verify unit test coverage
- Milestone 5: Let's assume documentation is written during development

## 7. Class Index

- `Zend_Microformat_Xfn`
- `Zend_Microformat_Xfn_Entity`
- `Zend_Microformat_Xfn_Result`
- `Zend_Microformat_Xfn_Exception`

## 8. Use Cases

Given XFN's simplistic data model, it is likely the steps needed to generate XFN links will also be abbreviated to a single `new()` method containing all three required parameters.

### UC-01

#### Fetch XFN data using default `Zend_Microformat` API

```
$mf = new Zend_Microformat('http://www.example.com');
$xfns = $mf->find('hcard');
foreach($xfns as $xfn) {
    echo 'Name: ', $xfn->name, PHP_EOL;
    echo 'Relations: ', array_implode(' ', $xfn->relation), PHP_EOL;
    echo 'Link: ', $xfn->link, PHP_EOL;
}
```

### UC-02

#### Fetch XFN data using `XFN` class directly

```
$mfxfn = new Zend_Microformat_Xfn;
$xfns = $mfxfn->find('http://www.example.com');
echo 'Found ', count($xfns), ' Entities' . PHP_EOL;
foreach($xfns as $xfn) {
    echo 'Name: ', $xfn->name, PHP_EOL;
    echo 'Relations: ', array_implode(' ', $xfn->relation), PHP_EOL;
    echo 'Link: ', $xfn->link, PHP_EOL;
}
```

### UC-03

#### Generating XFN embeddable output

```
$mfxfn = new Zend_Microformat_Xfn;
$xfn = $mfxfn->new();
$xfn->link = 'http://www.example.com/Joe';
$xfn->name = 'Joe Bloggs';
$xfn->relations = array('friend', 'met', 'neighbour');
echo $xfn->generate();
// <a href="http://www.example.com/Joe" rel="friend met neighbour">Joe Bloggs</a>
```

## 9. Class Skeletons

Please see the above use cases. Based on these, source code will be developed following Test-Driven Development and therefore class skeletons will not be presented in this proposal prematurely. The TDD process will implement `Zend_Microformat` specifically to enable the use cases presented above.

```
]]></ac:plain-text-body></ac:macro>
]]></ac:plain-text-body></ac:macro>
```