Basket : Session-based pseudo-mapper

The Basket plugin is a SESSION-based Object-Document-Mapper with a pretty similar syntax as the Cursor class (cursor) and its derivatives. This means that any addressed data is just available for the lifetime of the current user session, which makes it useful for shopping cart implementations or any service that works with guest visitors instead of registered members.

Namespace: `\`
File location: `lib/basket.php`

Instantiation

This is how you create a new Basket mapper:

```
$basket = new \Basket();
$basket_named = new \Basket( 'discounted-items' );
```

Please refer to the __construct (basket#__construct) method for details.

Methods

exists

Return TRUE if field is defined

```
bool exists ( string $key )
```

This function allows you to check if a named field is defined.

Example:

```
$basket->set('cherry',5);
$basket->exists('cherry'); // TRUE
```

set

Assign value to field

```
scalar|FALSE set ( string $key, scalar $val )
```

This function allows you to assign a value to a field.

Notice: The $key ' `_id` ' is a reserved key and would result in `set()` returning FALSE and not saving the key/value pair.

Returns the assigned value $val on success or FALSE if $key equals the reserved key ' `_id` '.

Example:

```
$val = $basket->set('product_code', 2764);
$val = $basket->set('product_discount', TRUE);
```

get

Retrieve value of field

```
scalar|FALSE get ( string $key )
```

This function allows you to retrieve the value of a field.

Example:

```
echo $basket->get('cherry'); // displays '5'
```

clear

Delete field

```
NULL clear ( string $key )
```

This function allows you to delete a field. The field won't exist anymore for this basket.

Example:

```
$basket->clear('cherry');
```

find

Return items that match key/value pair or all items when no key/value pair is specified

```
array|FALSE find ( [ string $key = NULL , mixed $val = NULL ] )
```

This function allows you to find items that match key/value pair. It returns an array of basket mapper objects.

If no key/value pair is specified, returns all the items of the basket. It gives you the ability to loop through the entire basket.

Example:

```
// add item
$basket->set('name','cherry');
$basket->set('amount',5);
$basket->save();
$basket->reset();
// add item
$basket->set('name','peach');
$basket->set('amount',10);
$basket->save();
$basket->reset();

$result = $basket->find('name','cherry');

echo $result[0]->get('amount'); // displays '5' (int)
echo $result[0]->get('name'); // displays 'cherry'

$result = $basket->find();  // array of basket items, if any

// results, e.g.:
array(2) {
  [0]=>
  object(Basket)#7 (3) {
    ["key":protected]=>
    string(5) "sunny"
    ["id":protected]=>
    string(23) "52db5405a64047.97371123"
    ["item":protected]=>
    array(2) {
      ["name"]   =>     string(6) "cherry"
      ["amount"] =>     int(5)
    }
  }
  [1]=>
  object(Basket)#8 (3) {
    ["key":protected]=>
    string(5) "sunny"
    ["id":protected]=>
    string(23) "52db5405a64143.07049859"
    ["item":protected]=>
    array(2) {
      ["name"]   =>     string(5) "peach"
      ["amount"] =>     int(10)
    }
  }
}
```

findone

Return first item that matches key/value pair

```
object|FALSE findone ( string $key, mixed $val )
```

This function allows you to get the first item that matches the given key/value pair.

Example:

```
$result = $basket->findone('amount','10');
echo $result->get('name'); // displays 'peach'
```

load

Map current item to matching key/value pair

```
array load ( string $key, mixed $val )
```

This function allows you to map current item to the matching key/value pair.

Example:

```
$basket->load('name','peach');
echo $basket->get('amount'); // displays '10' (int)
```

dry

Return TRUE if current item is empty/undefined

```
bool dry (  )
```

This function allows you to check if the current item is empty/undefined.

Example:

```
$basket->load('name','banana');
$basket->dry(); // TRUE, this mapper is empty

$basket->load('name','peach');
$basket->dry(); // FALSE, this mapper is hydrated
```

count

Return number of items in basket

```
int count (  )
```

This function allows you to return the number of items in the basket.

Example:

```
echo $basket->count(); // displays '2' (int)
```

save

Save current item

```
array save (  )
```

This function allows you to save the current basket item.

Example:

```
$basket->set('name','peach');
$basket->set('amount',10);
$basket->save();
```

erase

Erase item matching key/value pair

```
bool erase ( string $key, mixed $val )
```

This function allows you to erase the basket item matching the given key/value pair.

Example:

```
$basket->erase('name','peach'); // returns TRUE
```

reset

Reset cursor

```
NULL reset (  )
```

This function allows you to reset the cursor.

Example:

```
$basket->load('name','cherry'); // mapper is hydrated now
$basket->dry(); // false
$basket->reset();
$basket->dry(); // true, mapper is empty again
```

drop

Empty basket

```
NULL drop (  )
```

This function allows you to completely empty the basket contents. The basket is removed from the SESSION as well.

Example:

```
$basket->drop();
```

copyfrom

Hydrate item using array

```
NULL copyfrom ( array | string $var )
```

This function allows you to hydrate the basket using an array (or the name of a hive variable containing an array).

Example 1:

```
$basket->copyfrom(array('name'=>'banana', 'amount'=> 15));
$basket->get('name'); // banana
```

Example 2:

```
$f3->set('banana',array('name'=>'banana', 'amount'=> 15));
$basket->copyfrom('banana');
$basket->get('name'); // banana
```

copyto

Populate hive array variable with item contents

```
NULL copyto ( string $key )
```

This function allows you to populate a hive array variable with basket contents.

Example:

```
$basket->copyto($key);
```

checkout

Check out basket contents

```
array checkout (  )
```

This function allows you to check out basket contents, which means it returns all items and clears the whole basket.

Example:

```
$basket_items = $basket->checkout();
```

__construct

Instantiate class

```
void __construct ( [ string $key = 'basket' ] )
```

The constructor allows you to instantiate the class.

The `$key` parameter is an equivalent to a database table name.

Example:

```
$basket = new Basket( 'puppy-cart' );
```