Overview

Lexity Live® lets your merchants watch their customers shop their store, in real-time. This guide describes how you can integrate Lexity Live® into your offering easily.

In order to integrate Lexity Live®, you need to do the following 3 things:

1. **Acquire a shared key** from Lexity. This key will be used to ensure only you are accessing the Lexity Live® functionality.
2. **Embed the Lexity Live® code** on your merchants' home, product and checkout pages.
3. **Render the Lexity Live® iframe** in your merchants' admin interfaces.

1. Acquire a shared key

Email bizdev@lexity.com to acquire a partner code and shared key from Lexity. These will look something like this:

```
partner_code: se
shared_key: affeddfeddfe
```

You will use these in the next two steps.

2. Embed the Lexity Live® code

Insert the following code in your merchants' home, product and checkout pages, just before the `</head>` tag. Note the 3 dynamic parameters within this code, that you'll have to generate, in the following manner:

- `partner_code` is the one provided to you in Step 1
- `merchant_id` is a unique id, different for each merchant, that allows us to distinguish between each merchant on your platform. This doesn't need to be your internal merchant id (although it could be), it just needs to be different per merchant. The merchant id must satisfy a few requirements described in note **[1]**.
- `embed_hash` is the hex-digest of an md5 hash generated using the `merchant_id` and `shared_key`, in the manner below (see note **[2]**). This hash is used to ensure security.
    - `md5("e" + merchant_id + shared_key)`

Code:

```
<script type="text/javascript">
 (function (d, w) {
  var x = d.getElementsByTagName('SCRIPT')[0];
  var f = function () {
   var s = d.createElement('SCRIPT');
   s.type = 'text/javascript';
   s.async = true;
   s.src = "//np.lexity.com/embed/partner_code/embed_hash?id=merchant_id";
   x.parentNode.insertBefore(s, x);
  };
w.attachEvent ? w.attachEvent('onload',f) :
w.addEventListener('load',f,false);
 }(document, window));
</script>
```

Once this code is inserted on the merchant pages, Lexity will start receving the data it needs, in order to render the Lexity Live® UI for the merchants.

3. Render the Lexity Live® iframe

Insert the following code within your merchants' admin panels, wherever you would like the Lexity Live® UI to render. As in Step 2, there are a few dynamic parameters:

- `email` is the contact email for this merchant, URL-encoded (see note **[3]**).
- `store_url` is the URL for this merchant's home page, URL-encoded (see note **[3]**).
- `partner_code` is the same one provided in Step 1.
- `merchant_id` is the per-merchant unique id, and needs to be identical to the one used in Step 2 and must still satisfy the requirements described in note **[1]**.
- `render_hash` is the hex-digest of an md5 hash generated as below:
    - `md5("r" + merchant_id + shared_key)`

There are also a few static parameters:

- `width` is the width of the iframe in pixels, strongly recommended to be "`1024`".
- `height` is the height of the iframe in pixels, strongly recommend to be "`800`" or greater.

Code:

```
<iframe src="http://lexity.com/embed?
p=partner_code&h=render_hash&id=merchant_id&e=email&u=store_url"
height="height" width="width"></iframe>
```

Once this code is inserted on the merchant admin page, Lexity Live® will start appearing for your customers. Note that if Step 2 is not completed, there will be no data to show, and the UI will appear empty.

Lexity

Styling the Lexity Live® UI

The Lexity Live® UI can be styled simply by overriding a few CSS classes. Instructions on this will be provided in a future version of this document.

Notes

[1] The merchant id you provide to Lexity as `merchant_id` should be no more than 8 characters long. Furthermore, it should only include the letters a-z, A-Z or numbers 0-9. (The merchant id should match the regular expression `/^[0-9a-zA-Z]{1,8}$/`).

[2] To generate the hex digest of an MD5 hash of a string `str`, use the following functions:

Language	Function	Package
node.js	`require('crypto').createHash('md5').update(str).digest('hex')`	`crypto`

Python	`h = hashlib.md5()`
	`h.update('str')`
	`h.hexdigest()`
		`hashlib`

PHP	`md5(str)`	n/a
Ruby	`Digest::MD5.hexdigest(str)`	`digest/md5`

[3] When properly URL-encoded, “`store.com/welcome`" becomes "`store.com%2Fwelcome`" and "`name@provider.com`" becomes "`name%40provider.com`". To URL-encode a string `str`, use the following functions:

Language	Function	Package
Javscript	`encodeURIComponent(str)`	n/a
Python	`urllib.quote_plus(str)`	`urllib`
PHP	`urlencode`	n/a
Ruby	`CGI.escape(str)`	`cgi`