display behavior of the container site. Still, these kinds of modifications become important as the ecosystem of applications grows to mirror the look and feel of the container site.

Consider this FBML example:

```
<fb:flv src=http://fettermansbooks.com/newtitles.flv height="400"
  width="400" title="New Releases">
```

This translates to quite a long string of JavaScript, rendering a video play module; this element is controlled by Facebook, intentionally disallowing such behaviors as autoplay.

"Functionality package" tags

Some Facebook FBML tags encompass entire suites of common Facebook application functionality. `<fb:friend-selector>` creates a type-ahead friend selector package common to many Facebook pages, incorporating Facebook data (friends, primary networks), CSS styling, and JavaScript for keypress actions. Tags such as this enable the container site to encourage certain design patterns and elements of commonality among applications, as well as enable developers to quickly implement the behavior they would like.

FBML: A small example

Recall the improvements we were able to make to our hypothetical external website with the introduction of the `friends.get` and `users.getInfo` APIs to the original *http://fettermansbooks.com* code. Now we'll show an example of how FBML can combine the social data, privacy business logic, and feel of a fully integrated application.

If we were able to obtain the reviews of a book using a database call `book_get_all_reviews($isbn)`, we could combine friend data, privacy, and the "wall" feature to display reviews of the book using FBML on the container site through the code in Example 6-23.

EXAMPLE 6-23. Creating an application using FBML

```
// Wall-style social book reviews on Facebook
// FBML Tags used: <fb:profile-pic>, <fb:name>, <fb:if-can-see>, <fb:wall>

// from section 1.3
$facebook_friend_uids = $facebook_client->api_client->friends_get();
foreach($facebook_friend_uids as $facebook_friend) {
  if ($books_user_id = books_user_id_from_facebook_id($facebook_friend))
    $book_site_friends[] = $books_user_id;

}

// a hypothesized mapping, returning
// books_uid -> book_review object
$all_reviewers = get_all_book_reviews($isbn);

$friend_reviewers = array_intersect($book_site_friends, array_keys($all_reviewers));
```