jQuery 1.7 proposal: `jQuery.fn.on()`

Currently we have three different event API pairs: bind/unbind, live/die, and delegate/undelegate. Since they all use the same event lists and events under the covers, exposing the APIs separately can lead to incorrect expectations. For example #7520, this code will unbind not only the document's directly-bound click event, but the live handler for the links below it:

```
$("a").live("click", … );
$(document).bind("click", … );
$(document).unbind("click");
```

We could firewall the different event models, so that attaching an event through one of the event API trinity would not affect the other two, but I'd prefer to unify our event handling APIs so that it is clear the user is dealing with a single event model under the covers:

```
$(elems).on(events [, selector ] [, data ], handler);
```

A similar API signature was discussed for jQuery back in 2006 and Prototypejs has implemented a similar API recently.

All three existing APIs can be defined in terms of the new one for back-compat, though I suspect we will still support the old names forever:

```
$(elems).bind(events, fn) ⇒ $(elems).on(events, fn);
$(elems).delegate(events, selector, fn) ⇒ $(elems).on(events, selector, fn);
$(selector).live(events, fn) ⇒ $(document).on(events, selector, fn);
```

In addition to unifying the event API and documentation externally, this can also unify and DRY out the jQuery core event code. Right now there are very different code paths for live/delegate than for bind, although both paths need to support similar features (multiple event names separated by spaces, namespaces, remove all or just one function, etc). There is also some work done at event delivery time for delegated events that I think can be moved into event setup, improving event delivery performance.

It would be possible to update `.bind()` so that it does the same as the proposed new `.on()` API, but I think that would cause confusion -- all jQuery docs to this point talk about bind as non-delegated binding. However I think the existing signature for `.one()` can be extended to provide (optional) event delegation and go through the same code path; since it is less-used the potential for confusion is lower. Event removal of both delegated and directly-bound events can be exposed as `.un()` or `.off()`, for symmetry.

To summarize the advantages:

- A single API for event binding makes it clear that event delegation simply filters events for the given elements where the event is bound. Fixes the confusion expressed in #7520.
- Short but intuitive; the "on" prefix is naturally associated with events.
- Provides the opportunity to DRY out the event code.