Handling Redirects and Other Request Changes

A redirect occurs when a server responds to a request by indicating that the client should make a new request to a different URL. The `NSURLSession`, `NSURLConnection`, and `NSURLDownload` classes notify their delegates when this occurs.

To handle a redirect, your URL loading class delegate must implement one of the following delegate methods:

- For `NSURLSession`, implement the `URLSession:task:willPerformHTTPRedirection:newRequest:completionHandler:` delegate method.
- For `NSURLConnection`, implement the `connection:willSendRequest:redirectResponse:` delegate method.
- For `NSURLDownload`, implement the `download:willSendRequest:redirectResponse:` delegate method.

In these methods, the delegate can examine the new request and the response that caused the redirect, and can return a new request object through the completion handler for `NSURLSession` or through the return value for `NSURLConnection` and `NSURLDownload`.

The delegate can do any of the following:

- Allow the redirect by simply returning the provided request.
- Create a new request, pointing to a different URL, and return that request.
- Reject the redirect and receive any existing data from the connection by returning `nil`.

In addition, the delegate can cancel both the redirect and the connection. With `NSURLSession`, the delegate does this by sending the `cancel` message to the task object. With the `NSURLConnection` or `NSURLDownload` APIs, the delegate does this by sending the `cancel` message to the `NSURLConnection` or `NSURLDownload` object.

The delegate also receives the `connection:willSendRequest:redirectResponse:` message if the `NSURLProtocol` subclass that handles the request has changed the `NSURLRequest` in order to standardize its format, for example, changing a request for `http://www.apple.com` to `http://www.apple.com/`. This occurs because the standardized, or canonical, version of the request is used for cache management. In this special case, the response passed to the delegate is `nil` and the delegate should simply return the provided request.

The example implementation in Listing 5-1 allows canonical changes and denies all server redirects.

Listing 5-1 Example of an implementation of `connection:willSendRequest:redirectResponse:`

```
#if FOR_NSURLSESSION
- (void)URLSession:(NSURLSession *)session
        task:(NSURLSessionTask *)task
        willPerformHTTPRedirection:(NSHTTPURLResponse *)redirectResponse
        newRequest:(NSURLRequest *)request
        completionHandler:(void (^)(NSURLRequest *))completionHandler
#elif FOR_NSURLCONNECTION
-(NSURLRequest *)connection:(NSURLConnection *)connection
            willSendRequest:(NSURLRequest *)request
           redirectResponse:(NSURLResponse *)redirectResponse
#else // FOR_NSURLDOWNLOAD
-(NSURLRequest *)download:(NSURLConnection *)connection
```

```
             willSendRequest:(NSURLRequest *)request
            redirectResponse:(NSURLResponse *)redirectResponse
#endif
{
    NSURLRequest *newRequest = request;
    if (redirectResponse) {
        newRequest = nil;
    }

#if FOR_NSURLSESSION
    completionHandler(newRequest);
#else
    return newRequest;
#endif
}
```

If the delegate doesn’t provide an implementation for an appropriate redirect handling delegate method, all canonical changes and server redirects are allowed.