**Cross Site Scripting (XSS)**

Cross-Site Scripting (XSS) is a security vulnerability which enables an attacker to place client side scripts (usually JavaScript) into web pages. When other users load affected pages the attackers scripts will run, enabling the attacker to steal cookies and session tokens, change the contents of the web page through DOM manipulation or redirect the browser to another page. XSS vulnerabilities generally occur when an application takes user input and outputs it in a page without validating, encoding or escaping it.

**Protecting your application against XSS**

At a basic level XSS works by tricking your application into inserting a <script> tag into your rendered page, or by inserting an On\* event into an element. Developers should use the following prevention steps to avoid introducing XSS into their application.

1. Never put untrusted data into your HTML input, unless you follow the rest of the steps below. Untrusted data is any data that may be controlled by an attacker, HTML form inputs, query strings, HTTP headers, even data sourced from a database as an attacker may be able to breach your database even if they cannot breach your application.
2. Before putting untrusted data inside an HTML element ensure it's HTML encoded. HTML encoding takes characters such as < and changes them into a safe form like &lt;
3. Before putting untrusted data into an HTML attribute ensure it's HTML attribute encoded. HTML attribute encoding is a superset of HTML encoding and encodes additional characters such as " and '.
4. Before putting untrusted data into JavaScript place the data in an HTML element whose contents you retrieve at runtime. If this isn't possible then ensure the data is JavaScript encoded. JavaScript encoding takes dangerous characters for JavaScript and replaces them with their hex, for example < would be encoded as \u003C.
5. Before putting untrusted data into a URL query string ensure it's URL encoded.