Breaking Change: 4.10 Escaping by default in SpyWriter

Overview

Prior to Niagara 4.10, HTML could be passed to the td (), tr (), th (), trTitle (), thTitle (), a(), and prop() methods of SpyWriter, and that HTML would show up in the spy page of your Spy or BObject unaltered. This made it easier to introduce cross-site scripting (XSS) attacks if your spy passed user-entered data from the station directly into one of these SpyWriter methods. In Niagara 4.10, SpyWriter has been changed to always HTML-escape any values passed to these methods. This makes the majority of SpyWriter usage HTML-safe by default.

Remediation

Examine your own code for any of the following patterns in either an override of BObject#spy(SpyWriter) or Spy#write(SpyWriter):

Passing HTML directly to prop () or another affected method.

In most cases, simply look for other existing methods to accomplish your goals. To protect against XSS, make sure to make use of the safe () method when data can be provided by a user.

Existing code	New code
out.td (http://out.td) ("Text in bold")	out.td (http://out.td) ().w("Text in bold").endTd()
out.td (http://out.td) ("Other Page")	out.td (http://out.td) ().a("otherPage", "Other Page").endTd()
out.tr (http://out.tr) ("cell 1", "cell 2")	out.tr (http://out.tr) () .td().w("cell 1").endTd() .td().w("cell 2").endTd() .endTr()
out.w("" + userSuppliedValue + "")	out.w("").safe(userSuppliedValue).w("")
out.prop("Link from name column", "value")	out.propNameLink("myPage", "Link from name column", "value")
out.prop("name", "Link from value column")	out.propValueLink("name", "myPage", "Link from value column")

If a method does not already exist, use unsafe () to kick the SpyWriter into an unsafe mode where input will not be escaped. Only use this mode as long as necessary. Break the method chain to exit unsafe mode.

Existing code	New code
out.thTitle("Header") .trTitle("Section", 2) .tr("Cell 1", "Cell 2") .tr(userValue1, userValue2);	out.unsafe() .thTitle("Header") .trTitle("Section", 2) .tr("Cell 1", "Cell 2"); // break the chain to switch back to safe mode out.tr (http://out.tr) (userValue1, userValue2);

Passing unsafe values to w ()

Any value that is supplied by the user or read from the station, if passed directly to w (), represents a potential XSS vulnerability. This is because w () writes HTML directly to the page, without escaping. To mitigate this vulnerability, be sure to pass user-supplied data to safe ().

Existing code	New code

for (Slot slot : component.getSlotsArray())
{
out.w("Slot name: " + component.getDisplayName(slot, cx));
}
	for (Slot slot : component.getSlotsArray())
	{
	out.w("Slot name: ").safe(component.getDisplayName(slot, cx));
	}

Pre-escaping values passed to prop () or another affected method.

You may already be HTML-escaping values passed to one of these methods - great job! If so, remove the existing encoding - otherwise the value will double-encode, which will look strange in the browser.

Existing code	New code
out.td (http://out.td) (Encode.forHtml(userSuppliedValue))	out.td (http://out.td) (userSuppliedValue)
out.w("User value: " + Encode.forHtml(userSuppliedValue))	No change necessary – w () still does not apply any extra escaping.