Code Samples for Third-Party Analytics Integrations

You can use these sample code snippets with a variety of common third-party analytics vendors. While you can likely copy and paste these code examples following the steps in Configure a Custom Third-Party Analytics Report, note the modifications in some sections.

You may need to further customize a code sample to meet your individual needs. Submit a support ticket using the Kibo Technical Support portal (kibotechsupport.zendesk.com) if you need troubleshooting assistance.

Google Analytics 4

If you use Google tag (`gtag.js`) and need to customize your integration, use one of the code snippets in Set Up a Google Analytics 4 Integration Using Google Tag.

If you use Google Tag Manager, use the code snippet in Integrate with Google Analytics When Using Google Tag Manager.

Adobe Analytics (SiteCatalyst) with STL

This report code sends a track link to Adobe Analytics with the experiences that the site visitor saw. Because Adobe Analytics has limits and bills according to the number of server requests, this track line reporting may impact a client's contact with Adobe Analytics.
Because of how Adobe Analytics tracks bounce rate, this track line reporting affects the bounce rate analytics. To avoid this, use the integration without STL instead.
Modify the code sample by doing the following:

1. Replace the value of the `varKey` variable with the specific `eVar` or `listVar` to which labels should be sent.
2. Replace the value of the `opt_accountId` variable with the name of the report suite to which labels should be sent. One method for finding this is by looking at the window variable `s_i_xyz` where `xyz` is the unique report suite name.

```
var varKey = 'LIST_X';
var opt_accountId = 'ACCOUNT_ID';
var imgKey = 's_i_' + opt_accountId;

var opt_linkTrackEvents;
// set this if you want a delimiter other than | (should match what the client has set up if using
var opt_delimiter;

// translate array of objects into array of strings
var xr = [];
for (var i = 0, len = campaigns.length; i < len; i++) {
  var campaign = campaigns[i];
  xr.push(campaign.key + ':' + campaign.split);
```

```
}

var pollingFunction = function(interval, timeout, isReady, onReady, opt_onTimeout) {
  (function() {
    if (isReady()) {
      onReady();
    } else {
      if (timeout > 0) {
        timeout -= interval;
        setTimeout( /** @type {function()} */ (arguments.callee), interval);
      } else {
        if (opt_onTimeout) {
          opt_onTimeout();
        }
      }
    }
  })();
};

var submitToSiteCat = function(accountId, varKey, opt_linkTrackEvents, opt_delimiter) {
  var s = window.s_gi(accountId);
  if (s && s.tl) {
    var reportTxt = opt_delimiter ? xr.join(opt_delimiter) : xr.join('|');
    s.linkTrackVars = varKey + (opt_linkTrackEvents ? ',events' : '');
    if (opt_linkTrackEvents) {
      s.linkTrackEvents = opt_linkTrackEvents;
      s.events = s.linkTrackEvents;
    }
    s[varKey] = reportTxt;
    s.tl(true, 'o', 'Monetate');
  }
};

var pollForSiteCat = function(successHandler, imgKey, opt_accountId) {
  var pageViewComplete = function() {
    var pageViewPixel = window[imgKey];
    return pageViewPixel && pageViewPixel.src && pageViewPixel.complete;
  };

  pollingFunction(50, 10000, function() { // every 50ms for 10s
    return window.s_gi && window.Image && ((opt_accountId) ? true : window.s_account) &&
      pageViewComplete();
  }, function() {
    successHandler(opt_accountId || window.s_account);
  });
};
```

```
pollForSiteCat(function(accountId) {
  submitToSiteCat(accountId, varKey, opt_linkTrackEvents, opt_delimiter);
}, imgKey, opt_accountId);
```

Adobe Analytics (SiteCatalyst) Without STL

This report code doesn't directly send labels to Adobe Analytics. Instead, it's a generic one to store the information that would be reported in a cookie. It's especially useful for Adobe Analytics to avoid sending a track link that can adversely affect bounce rate metrics in Adobe Analytics.

> With this solution additional development is required on site to implement code that would take the information from the cookie and send it to the analytics provider however desired. It is usually delayed one-page load as the site usually reads the cookie at page load. The site is also responsible of clearing this cookie after the information is used.

```
/*
DEVELOPMENT NEEDED ON THE SITE:
Before s.t() is called (but after the SiteCatalyst object has been created), read the value of our
If it has a value, set the list variable 'list2' to the value in the cookie, then clear the cookie.
That's it! If it doesn't have a value, it wouldn't set the list variable and would just call s.t() as n
*/

// set this if you want a delimiter other than | (should match what the client has set up if using
var opt_delimiter;

// translate array of objects into array of strings
var xr = [];
for (var i = 0, len = campaigns.length; i < len; i++) {
  var campaign = campaigns[i];
  xr.push(campaign.key + ':' + campaign.split);
}

var reportTxt = opt_delimiter ? xr.join(opt_delimiter) : xr.join('|');
var cookieKey = 'mt.aa';

var getCookie = function(n, opt_spaces) {
  var cookies = (document.cookie || '').split(/\s*;\s*/);
  for (var i = 0, l = cookies.length; i < l; i++) {
    var cookie = cookies[i];

    // Index of the first equal sign, for cases where the cookie
    // value contains an equal sign. If you set a cookie with multiple
    // equal signs the value before the first equal sign is interpreted
    // as the name. example:
```

```
    // document.cookie = 'a=b';
    // document.cookie = 'a=b=c'; // overwrites a's value with 'b=c'
    var index = cookie.indexOf('=');

    // Skip Cookies with empty names.
    if (index != -1) {
      var name = cookie.substring(0, index);
      if (n === name) {

        // If there is no value this will be an empty string.
        var value = cookie.substring(index + 1);
        return unescape(opt_spaces ? value.replace(/\+/g, ' ') : value);
      }
    }
  }
  return null;
};

var setCookie = function(name, value, domain, path, expiration) {
  document.cookie = name + '=' + risonQuote(value) + '; ' +
    (domain && domain.length ? 'domain=' + domain + '; ' : '') + // empty domain string same
    (path && path.length ? 'path=' + path + '; ' : '') + // empty path string same as null
    (expiration ? 'expires=' + generateExpirationString(expiration) + '; ' : '');
};

var URI_OK = { // ok in url paths and in form query args
  '~': true,
  '!': true,
  '*': true,
  '(': true,
  ')': true,
  '-': true,
  '_': true,
  '.': true,
  ',': true,
  ':': true,
  '@': true,
  '$': true,
  "'": true,
  '/': true
};

var risonQuote = function(x) {
  // speedups todo:
  //   regex match exact set of URI_OK chars.
  //   chunking series of unsafe chars rather than encoding char-by-char
```

```
  if (/^[A-Za-z0-9_\-]*$/.test(x)) { // XXX add more safe chars
    return x;
  }

  x = x.replace(/([^A-Za-z0-9_\-])/g, function(a, b) {
    var c = URI_OK[b];
    if (c) {
      return b;
    }
    return encodeURIComponent(b);
  });
  return x.replace(/%20/g, '+');
};

var generateExpirationString = function(ms) {
  return new Date(new Date().getTime() + ms).toGMTString();
};

var cookieVal = getCookie(cookieKey, true);
if (cookieVal) {
  cookieVal += '|' + reportTxt;
} else {
  cookieVal = reportTxt;
}

setCookie(cookieKey, cookieVal, document.location.host, '/', 1800000);
```

IBM Coremetrics

This standard integration sends report labels as a conversion event and requires no modifications.

```
var counter = 0;
var interval = setInterval(function() {
  counter += 50;
  var done = false;

  if (window.cmCreateConversionEventTag && window._cm && window.Image) {
    var events = ''; // Combine all events into a pipe delimited string.
    var last = null;

    for (var i = 0, len = campaigns.length; i < len; i++) {
      var campaign = campaigns[i];
      if (events.length) {
        events += '|';
      }
      events += campaign.key + ',' + campaign.split;
    }

    if (events) {
      window.cmCreateConversionEventTag(events, 2, 'CGN - Monetate', 0);
    }
    done = true;
  }

  if (counter === 3000 || done) {
    clearInterval(interval);
  }
}, 50);
```

New Relic

This standard integration requires no modifications.

```
for (var i = 0; i < campaigns.length; i++) {
  newrelic.setCustomAttribute('MonetateExperienceID', campaigns[i].key);
  newrelic.setCustomAttribute('MonetateExperienceSplit', campaigns[i].split);
  newrelic.addPageAction('MonetateExperience');
}
```

Session Cam

This standard integration requires no modifications.

```
window.sessioncamConfiguration = window.sessioncamConfiguration || {};
window.sessioncamConfiguration.customDataObjects = window.sessioncamConfiguration.cust
for (var i = 0; i < campaigns.length; i++) {
  var item = {
    key: 'Monetate-' + campaigns[i].key,
    value: campaigns[i].split
  };
  window.sessioncamConfiguration.customDataObjects.push(item);
}
```

Clicktale

This is a standard integration and requires no modifications.

```
var interval = 50;
var timeout = 6000;
(function() {
  if (isReady()) {
    onReady();
  } else {
    if (timeout > 0) {
      timeout -= interval;
      setTimeout( /** @type {function()} */ (arguments.callee), interval);
    } else {
      if (opt_onTimeout) {
        opt_onTimeout();
      }
    }
  }
})();
};

var isReady = function() {
  return window.ClickTaleEvent &&
    window.ClicktaleIntegrationExperienceHandler;
};

var onReady = function() {
  var experiencesArray = [];
  // The `campaigns` variable is an array of experiences active on the page.
  for (var i = 0; i < campaigns.length; i++) {
    experiencesArray.push(campaigns[i].key + ',' + campaigns[i].split);
  }

  if (window.ClickTaleIsRecording && window.ClickTaleIsRecording()) {
    window.ClicktaleIntegrationExperienceHandler(experiencesArray);
  }
};

poll(isReady, onReady);
```

FullStory

This standard integration requires no modifications.

```
var _fs = function() {
  return window[window['_fs_namespace']];
};

var poll = function(isReady, onReady, opt_onTimeout) {
  var interval = 50;
  var timeout = 6000;
  (function() {
    if (isReady()) {
      onReady();
    } else {
      if (timeout > 0) {
        timeout -= interval;
        setTimeout( /** @type {function()} */ (arguments.callee), interval);
      } else {
        if (opt_onTimeout) {
          opt_onTimeout();
        }
      }
    }
  })();
};

var isReady = function() {
  return typeof _fs() === 'function';
};

var onReady = function() {
  var experiencesArray = [];
  // The `campaigns` variable is an array of experiences active on the page.
  for (var i = 0; i < campaigns.length; i++) {
    payload = {};
    payload.experiment = {};
    payload.experiment.id_str = campaigns[i].key;
    payload.experiment.name_str = campaigns[i].split;
    _fs().event('Experiment', payload, 'Monetate');
  }
};

poll(isReady, onReady);
```

Glassbox

This standard integration requires no modifications.

```
var counter = 0;
var interval = setInterval(function () {
  if (typeof (window._detector) === "object") {
    clearInterval(interval);
    for (var i = 0; i < campaigns.length; i++) {
      window._detector.triggerABTestingEvent("Monetate", campaigns[i].id, campaigns[i].key, ca
    }
    if (counter++ > 30) {
      clearInterval(interval);
    }
  }
}, 100);
```