

API

- API
 - Chart ID
 - ID
 - - name
 - reportUUID
 - entityUUID
 - filters
 - fields
 - charts
 - reportKey
 - API
 - useDefaultAssistedInsightsPanel
 - preventDefaultAssistedInsights
- - getFieldId(fieldName)ID
 - getFieldName(fieldId)
 - getChartId(chartId)ID
 - applySeriesSelection(series, chartId)
 - createReportElement(elementOptions)
 - -
 - sortAscending(fieldId)
 - sortDescending(fieldId)
 - runReport()
 - runAssistedInsights
 - -
 - openComparePanel(value, comparableValues, metricDescription)
 - displayAssistedInsightsData(assistedInsightsData)
 - cancelAssistedInsights(assistedInsightsUUID)
 - deleteTemporaryAssistedInsightsReport(assistedInsightsReportId)
 - registerOutputType(outputOptions, callback)
 - removeOutputType(outputKey)
 - reset()
 - isDrillAnywhere()
 - isDrillDown()
 - isDrillDownApplied()
 - isDrillAnywhereApplied()
 - drill(fieldId, value, toField)
 - drillUpLevels(fieldId, levels)
 - drillUpOneLevel()1
 - drillReset(fieldId)
 - timeSlider(from, to, chartId)
 - unitSelection(unit, chartId)
 - convertDrillDownToArray()&convertDrillAnywhereToArray()
- - seriesSelection
 - sort
 - reportStart
 - reportComplete
 - assistedInsightsRequestedByChart
 - assistedInsightsStarted
 - compareValuesSelected
 - assistedInsightsInProgress
 - assistedInsightsUpdate
 - assistedInsightsCompleted
 - assistedInsightsCancelled
 - assistedInsightsError
 - assistedInsightsExceptionError
 - drilldown
 - drillanywhere
 - drillUp
 - reportRunAlreadyInProgress
 - reset
 - slider
 - unitSelection

API

API

APIYellowfin API

Chart IDChart ID

chartIdchartUUIDchartIdchartUUIDchartIdchartUUIDchartIdchartUUID
chartId

ID

IDIDgetFieldId(fieldName)IDfieldIdfieldId

- 1.
- 2.
- 3.

YellowfinAgency TypeAgency NameAgency TypeAgency Name

SQL

```
SELECT DISTINCT
  "TRAVELAGENCY"."AGENCYTYPE" AS C1
FROM "TRAVELAGENCY"
```

AgencySQLAgency TypeAgency Type

```
SELECT DISTINCT
  "TRAVELAGENCY"."AGENCYNAME" AS C1
FROM "TRAVELAGENCY"
WHERE (
  "TRAVELAGENCY"."AGENCYTYPE" = 'Agency'
)
```

API

name

String

reportUUIDUUID

String

UUID

entityUUIDUUID

String

null

filters

[API](#)

API API API

fields

Array[ReportField]

charts

Object{String, ReportChart}

UUID

reportKey

String

API

API

useDefaultAssistedInsightsPanel

true

displayAssistedInsightsDatafalse

```
let reportElement = this.apis.canvas.select('Performance by Region');
reportElement.onReportLoad.then(() => {
  let reportAPI = reportElement.reportAPI;
  reportAPI.useDefaultAssistedInsightsPanel = true;
});
```

preventDefaultAssistedInsights

false

runAssistedInsights

UI

```
let reportElement = this.apis.canvas.select('Performance by Region');
reportElement.onReportLoad.then(() => {
  let reportAPI = reportElement.reportAPI;
  reportAPI.preventDefaultAssistedInsights = true;
});
```

getFieldId(fieldName)ID

Number

fieldNameIDnull

InvoicedID

```
let fieldId = report.getFieldId('Invoiced');
```

getFieldName(fieldId)

String

fieldIdnull

```
report.addListener('drilldown', function(event) {
  console.log(report.getFieldName(event.eventData.fieldId) + " just had a drill value applied to it");
})
```

getChartId(chartId)ID

Number

UUIDIDIDIDnull

applySeriesSelection(series, chartId)

ID

シリーズ選択を表示
グラフ上にどのシリーズを表示するかを表示ページで選択することができます。

☒

シリーズ選択のスタイル
シリーズのプレビューとキーとなる情報が表示され、グラフの上のパネルからシリーズを選択することができます。

シリーズ選択を表示 ☐

左パネル ☐

トップパネル ☒

fieldIdapplySeriesSelectionfieldId

Series - Array[Number]

ID

chartId - String, Number

IDIDID

```
let fieldsToAdd = ['Invoiced', 'Cost', 'Profit'];
report.applySeriesSelection(fieldsToAdd );
```

ID

```
report.applySeriesSelection(5);
```

```
report.applySeriesSelection('Invoiced');
```

createReportElement(elementOptions)

HTML

API

elementelementelementOptionselement



[BaseAPI.loader](#)reportId, instanceNamefilterValuescreateReportElement

1element

HTML

```
<div id="reportContainer"></div>
```

CSS

```
div.reportContainer app-report {
  width:500px;
  height:500px;
}
```

JS

```
let reportUUID = 'a-report-uuid';
yellowfin.loadReport({
  reportId: reportUUID
}).then(report => {
  let charts = report.charts;
  Object.values(charts).forEach(chart => {
    report.createReportElement({
      element: document.querySelector("#reportContainer"),
      displayType: 'CHART', //Tell the report to render as a chart
      chartUUID: chart.uuid, //Tell the report which chart you wish to render
      displayToolbar: false //Tell the report not to render any toolbar
    });
  });
});
```

500×500

2

HTML

```
<div id="reportContainer"></div>
```

CSS

```
div.reportContainer app-report {
  width:500px;
  height:500px;
}
```

JS

```
let reportUUID = 'a-report-uuid';
yellowfin.loadReport({
  reportId: reportUUID
}).then(report => {
  report.createReportElement({
    element: document.querySelector('#reportContainer'),
    displayType: 'CANVAS', //Tell the report to render as a chart
    displayToolbar: false //Tell the report not to render any toolbar
  });
  report.createReportElement({
    element: document.querySelector('#reportContainer'),
    displayType: 'REPORT', //Tell the report to render as a chart
    displayToolbar: false //Tell the report not to render any toolbar
  });
});
```

Yellowfin

elementelementelementelement

false

animation

annotations

brushing

drillAnywhere

drillBreadcrumbs

drillDown

drillThrough

unitSelection

timeSlider

seriesSelection

annotations

creatReportElement

```
report.createReportElement({
  interactions: {
    animation: false,
  },
  annotations: false,
  brushing: false,
  drillBreadcrumbs: false,
  drillDown: false,
  drillAnywhere: false,
  drillThrough: false,
  unitSelection: false,
  timeSlider: false,
  seriesSelection: false,
})
```

sortAscending(fieldId)

ID

fieldId: Number, String

ID

sortDescending(fieldId)

ID

fieldId: Number, String

ID

runReport()

runReport

5

```
//Add a listener for the reportComplete event
report.addListener('reportComplete', () => {
  //Add a 5000ms delay before triggering runReport again
  setTimeout(() => {
    report.runReport();
  }, 5000)
});

report.runReport();
```

runAssistedInsights

Promise

PromiseassistedinsightsCompleted

assistedInsightsRequestedByChart eventData

Options{object}

- loader{DOM }
- reportTitle{String}

options object

- type{String} - compareexplain
- metricField
- metricDescription
- categoryField
- value1{String Number}:

explainString

- dateValue
- timeSeries
- discreteTimeSeries
- granularity

compareString

- categoryFieldId
- categoryKey
- value2{String Number}:
- valueOneFormattedvalue oneAustralia

- valueTwoFormattedvalue two

- timeSeries
- discreteTimeSeries
- timeGranularity

runAssistedInsightsassistedInsightsRequestedByChart

```
let assistedInsightsData = {
  categoryField: 60629,
  categoryFieldId: 1,
  categoryKey: "61039",
  dashboardName: "Sales Performance",
  dashboardUUID: "e7409ff2-f846-44e1-a603-b78ec51b20b9",
  metricDescription: "KPI",
  metricField: 60686,
  reportTitle: "Comparing KPI for Europe to Asia",
  parentReportId: 61035,
  sourceId: 54700,
  type: "compare",
  value1: "Europe",
  value2: "Asia",
  valueOneFormatted: "Europe",
  valueTwoFormatted: "Asia",
  viewId: 60543
};
reportAPI.runAssistedInsights(assistedInsightsData);
```

openComaprePanel(value, comparableValues, metricDescription)

compareValuesSelected compareValuesSelected
openComaprePanelpreventDefaultAssistedInsightstruerunAssistedInsightsopenComaprePanel

- value{Object}: valueObject
 - value/rawValuedescription/formattedValue

```
{
  value: "EU",
  description: "Europe"
}
```

```
{
  rawValue: "EU",
  formattedValue: "Europe"
}
```

 assistedInsightsRequestedByChart valueeventDatavalue 1comparableValuespossibleValues

- comparableValues{Array}: valueObjects

```
[[
  {
    value: "AUS",
    description: "Australia"
  },
  {
    value: "NA",
    description: "North America"
  }
]]
```

- metricDescription{String}KPIUI

```
reportAPI.addListener('assistedInsightsRequestedByChart', event => {
  let runAssistedInsightsData = event.eventData;
  if (runAssistedInsightsData.type === 'compare' && runAssistedInsightsData.value2 == null) {
    reportAPI.openComparePanel(runAssistedInsightsData.value1, runAssistedInsightsData.possibleValues, runAssistedInsightsData.metricDescription);
  }
});
```

displayAssistedInsightsData(assistedInsightsData)

APIuseDefaultAssistedInsightsPanelfalse

deleteTemporaryAssistedInsightsReport

```
let runAssistedInsightsPromise = reportAPI.runAssistedInsights(data);
runAssistedInsightsPromise.then(assistedInsightsData => {
  // Display the report results
  reportAPI.displayAssistedInsightsData(assistedInsightsData);
});
```

cancelAssistedInsights(assistedInsightsUUID)

UUIDassistedInsightsProgress eventData

```
reportAPI.addListener('assistedInsightsInProgress', event => {
  let assistedInsightsTaskUUID = event.eventData;
  reportAPI.cancelAssistedInsights(assistedInsightsTaskUUID);
});
```

deleteTemporaryAssistedInsightsReport(assistedInsightsReportId)

Yellowfin

- assistedInsightsReportId{Number}: IDrunAssistedInsights promiseassistedInsightsCompleted

```
let runAssistedInsightsPromise = reportAPI.runAssistedInsights(data);
runAssistedInsightsPromise.then(assistedInsightsData => {
  let reportId = assistedInsightsData.assistedInsightsReportId;
  reportAPI.deleteTemporaryAssistedInsightsReport(reportId);
});
```

registerOutputType(outputOptions, callback)

Number

outputType
reportComplete
Yellowfin 9.2

- **rawValue**
- **formattedValue**
- **htmlFormattedValueHTML**

Gender

```
{
  rawValue: 'FEMALE', //In the SkiTeam database the Gender code is all caps
  formattedValue: 'Female', //After it has been formatted as a refcode it becomes a more readable version
  htmlFormattedValue: 'Female' //For this case, they are exactly the same.
}
```

formattedValuehtmlFormattedValue

formattedValuehtmlFormattedValueHTMLGender As LinkURL

Gender	Gender As Link
Female	FEMALE

```
[[
  {
    formattedValue: "Female"
    htmlFormattedValue: "Female"
    rawValue: "FEMALE"
  },
  {
    formattedValue: "FEMALE"
    htmlFormattedValue: "<a href='localhost:8080/myTestSite/FEMALE' target='_blank' rel='nofollow noopener norereferrer'>FEMALE</a>"
    rawValue: "FEMALE"
  }
]]
```

htmlFormattedValueformattedValue

outputOptions: String, Object

ObjectStringString

callback: Function

```
report.registerOutputType('dataset', function(reportDataset) {
  console.log(reportDataset);
});
```

outputInformation

```
report.registerOutputType({ resultType: 'dataset' }, function(reportDataset) {
  console.log(reportDataset);
});
```

removeOutputType(outputKey)

outputKeyYellowfin

outputKey: Number

outputKey

```
let outputId = report.registerOutputType('dataset', function(data) {
  console.log(data);
  report.removeOutputType(outputId);
});
```

reset()

filter.resetFiltersToDefault()filters.clear()

```
let resetButton = document.querySelector('div#resetReportButton');
resetButton.addEventListener('click', function(e) {
  report.reset();
});
```

Add a reset button that can reset the reports interactions and its filters back to their default values

```
let resetButton = document.querySelector('div#resetReportButton');
resetButton.addEventListener('click', function(e) {
  report.reset();
  report.filters.resetFiltersToDefault();
});
```

isDrillAnywhere()

Boolean

分析スタイル ^

- ☐ マルチ表示
- ☐ ドリルなし
- ☐ ドリルダウン
- ☒ ドリルエニウェア
- ☐ ドリルスルー

isDrillDown()

Boolean

分析スタイル ^

- ☐ マルチ表示
- ☐ ドリルなし
- ☒ ドリルダウン
- ☐ ドリルエニウェア
- ☐ ドリルスルー

isDrillDownApplied()

Boolean

```
console.log(report.isDrillDownApplied()); //Will return "false" as no drilling has been applied
report.drill(1, 'Agency');
console.log(report.isDrillDownApplied()); //Will now return "true" now that has drill down has been applied
```

isDrillAnywhereApplied()

Boolean

drill(fieldId, value, toField)

ID
toFieldfieldIdtoField

fieldId: Number, String

value: Number, String

toField: Number

ID

IDAgency TypeAgency

report.drill(1, 'Agency');

ID

report.drill('Agency Type', 'Agency');

drillUpLevels(fieldId, levels)

fieldId
report.drillUpLevels(fieldId, 1)
report.drillUpLevels(fieldId, 5)

field: Number, String

levels: Number

fieldId 1

report.drillUpLevels(1, 1);

drillUpOneLevel()1

1

```
let drillUpButton = document.querySelector('div#drillUp');
drillUpButton.addEventListener('click', function() {
  report.drillUpOneLevel();
});
```

drillReset(fieldId)

fieldIdfield Id

fieldId: Number, String

```
report.drillReset();
```

Agency Region

```
report.drillReset('Agency Region');
```

IDAgency Region

```
report.drillReset(1);
```

timeSlider(from, to, chartId)

fromtoID

from: Number

to: Number

chartId: Number, String

ID

August 201420148November 2014201411

```
//Create the date objects for the days we care about and get their time values
let fromDate = new Date('2014-08-01').getTime();
let toDate = new Date('2014-11-01').getTime();
report.timeslider(fromDate, toDate);

//Set the slider value based on the ChartUUID
let fromDate = new Date('2014-08-01').getTime();
let toDate = new Date('2014-11-01').getTime();
report.timeslider(fromDate, toDate, '0b808dd1-2114-42bc-a358-5fe7bf2ec052');
```

unitSelection(unit, chartId)

ID

unit: String

- MILLSECOND
- SECOND
- MINUTE
- HOUR
- DAY
- WEEK
- MONTH
- YEAR

MONTH

```
report.unitSelection('MONTH'); //No chart Id passed so the default chart will be used

let chartUUID = 'b779c293-a8ac-44cb-82f5-0c64da385333';

report.unitSelection('MONTH', chartUUID ); //Apply with a specific chart uuid

let chartName = 'Chart One';

report.unitSelection('MONTH', chartName); //Pass the chart name to determine which chart to use
```

convertDrillDownToArray()&convertDrillAnywhereToArray()

Array[Object]

- fieldId
- reportIdID
- reportKey
- reportUUIDUUID
- templateIdID
- value

2NN

123

- eventData
- matadeta
 - reportIdID
 - reportUUIDUUID
 - reportKey
- eventName

seriesSelection

- series(Array[Number])ID
- chartIdID

sort

- fieldId
- sortDirectionASCENDINGDESCENDING

reportStart

reportComplete

registerOutputType

```
{ //Example Empty Dataset
  8399975157147: [//A random ID for the output type
    [] //Report Dataset data
  ]
}
```

assistedInsightsRequestedByChart

useDefaultAssistedInsightsPanelfalse

runAssistedInsights

eventData

runAssistedInsights

eventData

- categoryField
- categoryFieldId
- categoryKey
- dashboardName:
- dashboardUUID: UUID
- metricDescription
- metricField
- parentReportId: ID
- possibleValuesvalue 1
- reportKey
- sourceIdparentReportID
- typeexplaincompare
- value1
- value2
- viewIdparentReportID

```
reportAPI.addListener('assistedInsightsRequestedByChart', chartData => {

  // This already contains all of the data we need to run Assisted Insights so we don't need to make any additional changes to it

  let newData = Object.assign({}, chartData.eventData);

  let promise = this.reportAPI.runAssistedInsights(newData);

});
```

assistedInsightsStarted

runAssistedInsights

```
reportAPI.addListener(assistedInsightsStarted, () => {  
  console.log("Assisted Insights has started");  
});
```

compareValuesSelected

eventData

- value1{StringNumber}: value oneopenComparePanelvalue/rawValue
- valueOneFormatted{StringNumber}: value oneopenComparePaneldescription/formattedValue
- value2{StringNumber}: value twonopenComparePanelvalue/rawValue
- valueTwoFormatted{StringNumber}: value twoopenComparePaneldescription/formattedValue

```
this.reportAPI.addListener('compareValuesSelected', event => {  
  let comparisonValues = event.eventData;  
  // Combine them with the existing Assisted Insights data  
  let allData = Object.assign({}, assistedInsightsData, comparisonValues);  
  // Generate the assisted insights report  
  reportAPI.runAssistedInsights(allData);  
});
```

assistedInsightsInProgress

eventData

- UUIDcancelAssistedInsights

UUIDUUIDrunAssistedInsights promiseassistedInsightsCompleted ID

```
reportAPI.addListener('assistedInsightsInProgress', event => {  
  let assistedInsightsTaskUUID = event.eventData;  
  reportAPI.cancelAssistedInsights(assistedInsightsTaskUUID);  
});
```

assistedInsightsUpdate

eventData

- progressText{String};
- state{String};

```
reportAPI.addListener('assistedInsightUpdate', event => {  
  let progressData = event.eventData;  
  let customLoader = document.getElementById("myCustomLoader").  
  customLoader.innerHTML(progressData.progressText);  
});
```

assistedInsightsCompleted

runAssistedInsightsprocess

runAssistedInsights promise

eventData

- assistedInsightsReportId{Number}: IDdeleteTemporaryAssistedInsightsReport
- chartData{Object}:
 - assistedInsightsImage{base64 String};
 - assistedInsightsZoomedImage{base64 String};
 - chartId{Number}: ID
 - narratives{Array};

```
reportAPI.addListener('assistedInsightsCompleted', data => {  
  // Display the report results  
  reportAPI.displayAssistedInsightsData(data.eventData);  
});
```

assistedInsightsCancelled

cancelAssistedInsights

eventData

UUID

```
reportAPI.addListener('assistedInsightsCancelled', event => {  
  console.log("Task " + event.eventData + " has been cancelled");  
});
```

assistedInsightsError

eventData

- getMessageText{function};

```
reportAPI.addListener('assistedInsightsError', event => {  
  console.log(event.eventData.getMessageText());  
});
```

assistedInsightsExceptionError

eventData

- errorMessage:

```
reportAPI.addListener('assistedInsightsExceptionError', event => {  
  console.log(event.eventData.errorMessage);  
});
```

drilldown

- **fieldId**
- **added**(Array)
- **drillState**(Array)convertDrillDownToArray

drillanywhere

- **added**(Array)
- **drillState**(Array)convertDrillAnywhereToArray

drillUp

- **removed**(Array)
- **drillState**(Array)convertDrillAnywhereToArrayconvertDrillDownToArray

reportRunAlreadyInProgress

```
report.addEventListener("reportRunAlreadyInProgress", function() {  
  alert("report is already running");  
});
```

resetDrill

- **removed**(Array)
- **drillState**(Array)convertDrillAnywhereToArrayconvertDrillDownToArray

reset

slider

- **from**(Number)
- **To**(Number)
- **chartId**(Number)

unitSelection

- **unit**(String)
- **chartId**(Number)