



Lightning Component Best Practices

introhive  **MapAnything™**  **getfeedback**



Lightning Component Best Practices

Sara Morgan Nettles, Lead Technical Curriculum Engineer at Salesforce

saramorgan.net, @SaraHasNoLimits



Safe Harbor

Safe harbor statement under the Private Securities Litigation Reform Act of 1995:

This presentation may contain forward-looking statements that involve risks, uncertainties, and assumptions. If any such uncertainties materialize or if any of the assumptions proves incorrect, the results of salesforce.com, inc. could differ materially from the results expressed or implied by the forward-looking statements we make. All statements other than statements of historical fact could be deemed forward-looking, including any projections of product or service availability, subscriber growth, earnings, revenues, or other financial items and any statements regarding strategies or plans of management for future operations, statements of belief, any statements concerning new, planned, or upgraded services or technology developments and customer contracts or use of our services.

The risks and uncertainties referred to above include – but are not limited to – risks associated with developing and delivering new functionality for our service, new products and services, our new business model, our past operating losses, possible fluctuations in our operating results and rate of growth, interruptions or delays in our Web hosting, breach of our security measures, the outcome of any litigation, risks associated with completed and any possible mergers and acquisitions, the immature market in which we operate, our relatively limited operating history, our ability to expand, retain, and motivate our employees and manage our growth, new releases of our service and successful customer deployment, our limited history reselling non-salesforce.com products, and utilization and selling to larger enterprise customers. Further information on potential factors that could affect the financial results of salesforce.com, inc. is included in our annual report on Form 10-K for the most recent fiscal year and in our quarterly report on Form 10-Q for the most recent fiscal quarter. These documents and others containing important disclosures are available on the SEC Filings section of the Investor Information section of our Web site.

Any unreleased services or features referenced in this or other presentations, press releases or public statements are not currently available and may not be delivered on time or at all. Customers who purchase our services should make the purchase decisions based upon features that are currently available. Salesforce.com, inc. assumes no obligation and does not intend to update these forward-looking statements.



I Am Also a Pluralsight Author



- [Getting Started Building SPAs with Lightning Components](#)
- [Customizing Salesforce with Lightning Components](#)
- [Lightning Component Development Best Practices](#)



PLURALSIGHT



Why Should I Care About Performance?



In the Mobile
World Every
Millisecond Counts



6 – Check your Settings



Enable/Disable Component Caching:

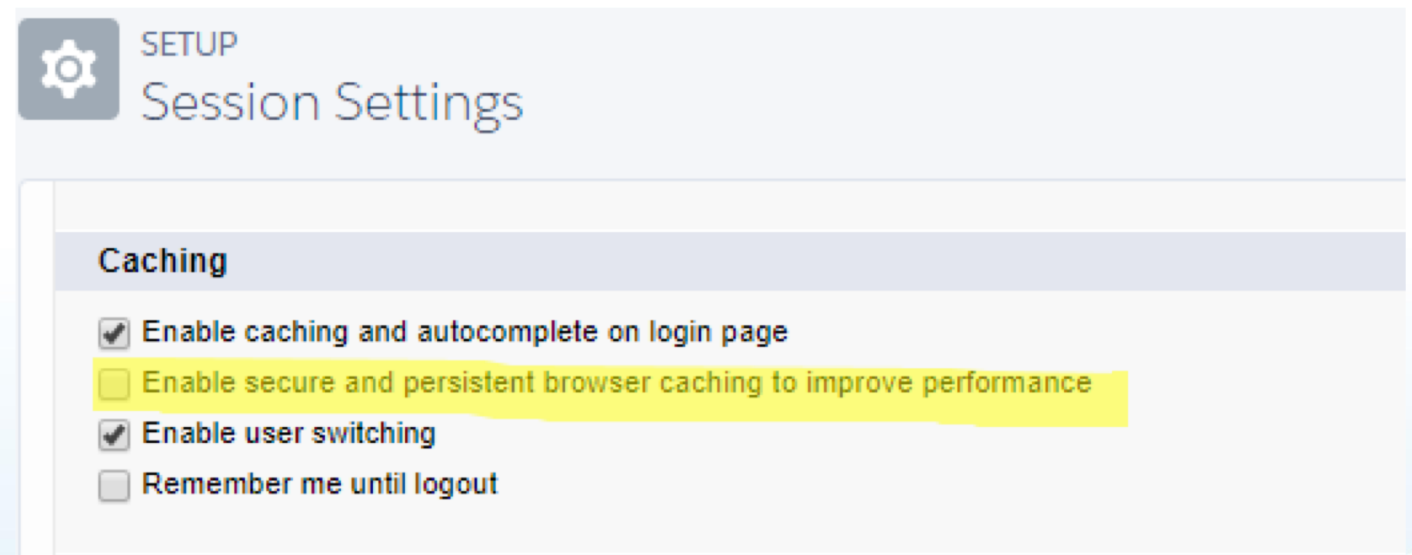
1. From Setup, go to **Session Settings**.
2. Select/Deselect the **Enable secure and persistent browser caching to improve performance** checkbox.
3. Click **Save**.



Definitely Enable
for Production



May Want to Disable for
Development



6 – Check your Settings



```
{ } project-scratch-def.json •
1  {
2    "orgName": "sara.morgan Company",
3    "edition": "Developer",
4    "orgPreferences" : {
5      "enabled": ["S1DesktopEnabled"],
6      "disabled": ["S1EncryptedStoragePref2"]
7    }
8  }
9  |
```

For SalesforceDX developers, you can disable the setting automatically for scratch orgs with the following line in your config file



6 – Check your Settings



To Enable/Disable Debug Mode:

1. From Setup, go to **Debug Mode**.
2. Enable/Disable the checkbox for a specific user.



ONLY Enable for
Development or
Sandboxes

Debug Mode Users

Enable debug mode to make it easier to debug JavaScript code from Lightning components. Only enable debug mode for users who have debug mode enabled.

View: All [Create New View](#)

A | B | C | D | E |

<div>Enable Disable</div>				
<input type="checkbox"/>	Full Name ↑	Debug Mode	Alias	Username
<input type="checkbox"/>	Chatter Expert	<input type="checkbox"/>	Chatter	chatty.00d410000011enbeaa.pteqzafefiug@chatter.salesforce.com
<input type="checkbox"/>	Nettles, Sara	<input checked="" type="checkbox"/>	SNett	lightning2@synapticap.com
<input type="checkbox"/>	User, Integration	<input type="checkbox"/>	integ	integration@00d410000011enbeaa.com
<input type="checkbox"/>	User, Security	<input type="checkbox"/>	sec	insightssecurity@00d410000011enbeaa.com



Make sure you Disable
for Production

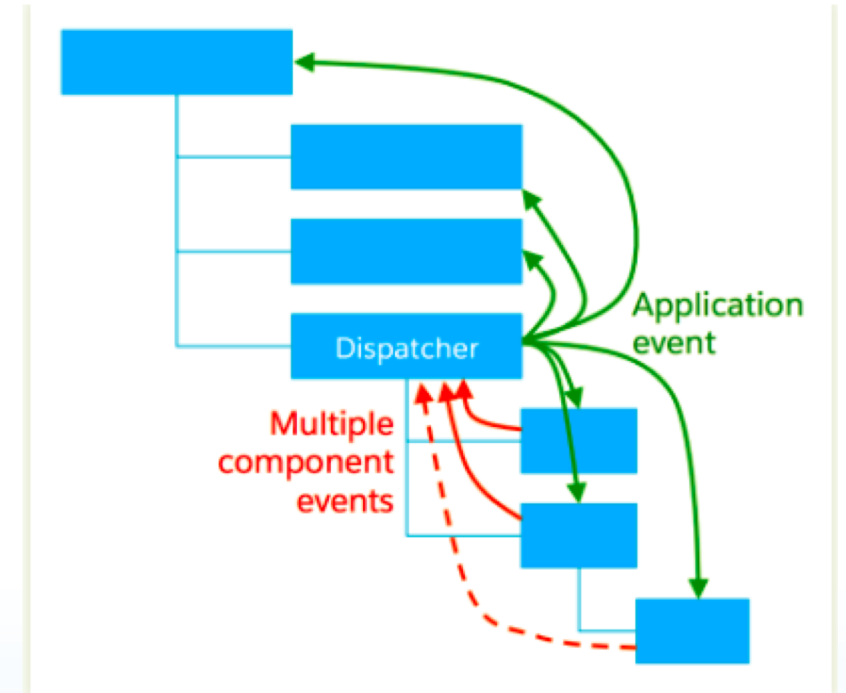
5 - Limit Use of Application Events



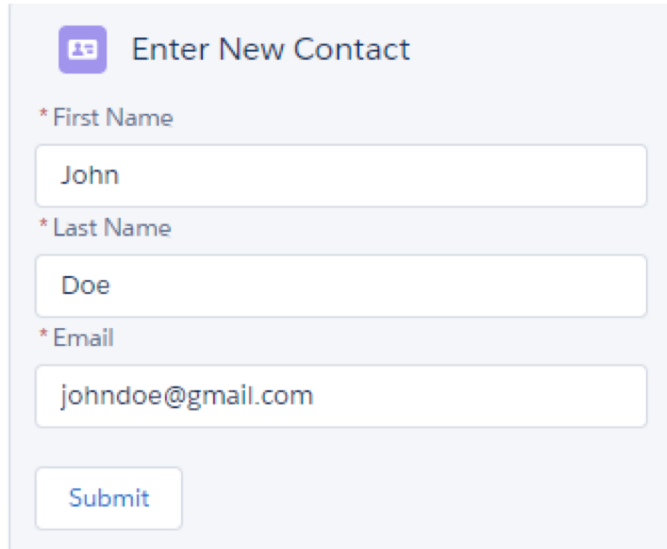
Consider using Component Events before Application events

Learn more about Inter-Component Communication Patterns

<https://sforce.co/2nYO1pK>



4 – Use <aura:if> for Conditional Rendering



Enter New Contact

* First Name
John

* Last Name
Doe

* Email
johndoe@gmail.com

Submit

```
<div aura:id="submitBtn" class="slds-hide">
```

```
<lightning:button label="Submit"
```

```
onclick="{!c.newRecord}" />
```

```
</div>
```

```
cmp.find("submitBtn").toggleClass(!isVisible, "slds-hide");
```



Don't Use this

4 – Use <aura:if> for Conditional Rendering

```
<aura:if isTrue="{!v.isVisible}">
```

```
    <lightning:button label="Submit"  
        onclick="{!c.newRecord}" />
```

```
</aura:if>
```

← Use this instead

```
cmp.set("v.isVisible", true);
```



3 – Use the Lightning Data Service When Possible

AKA: The Standard Controller for Lightning

Benefits include:

No APEX or SOQL needed

FLS and CRUD security built-in

Auto-notify on record changes

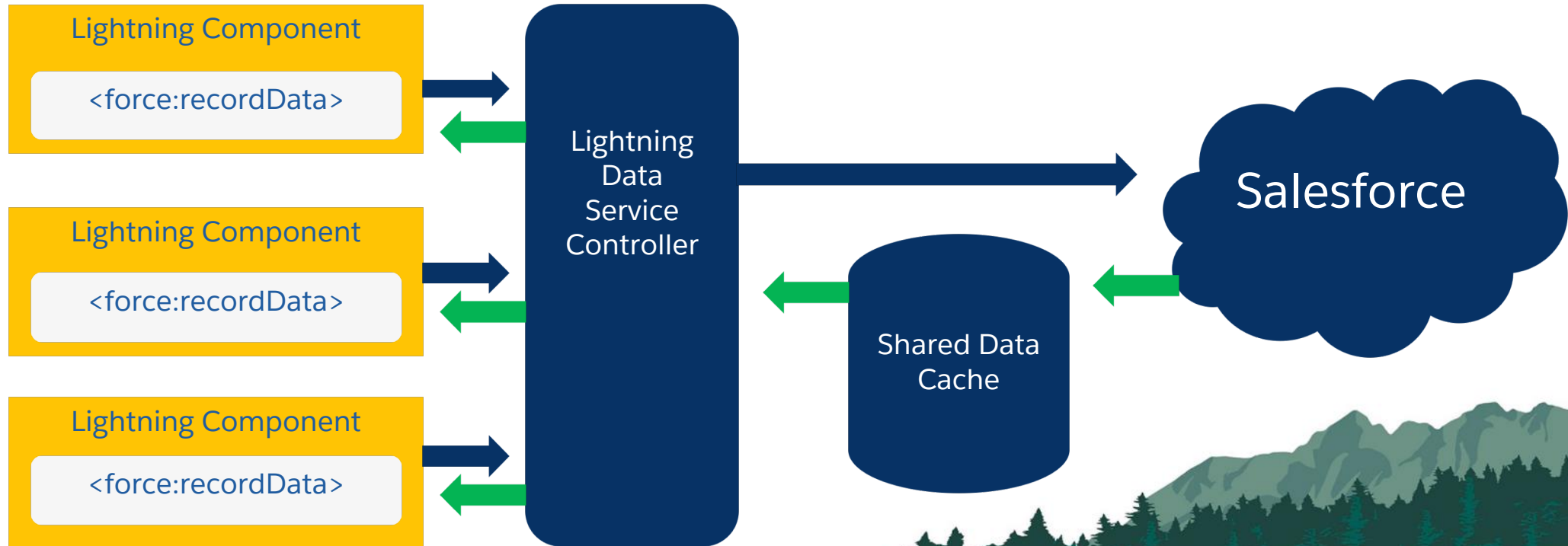
Offline access for Salesforce1

Single Request and Cached Response



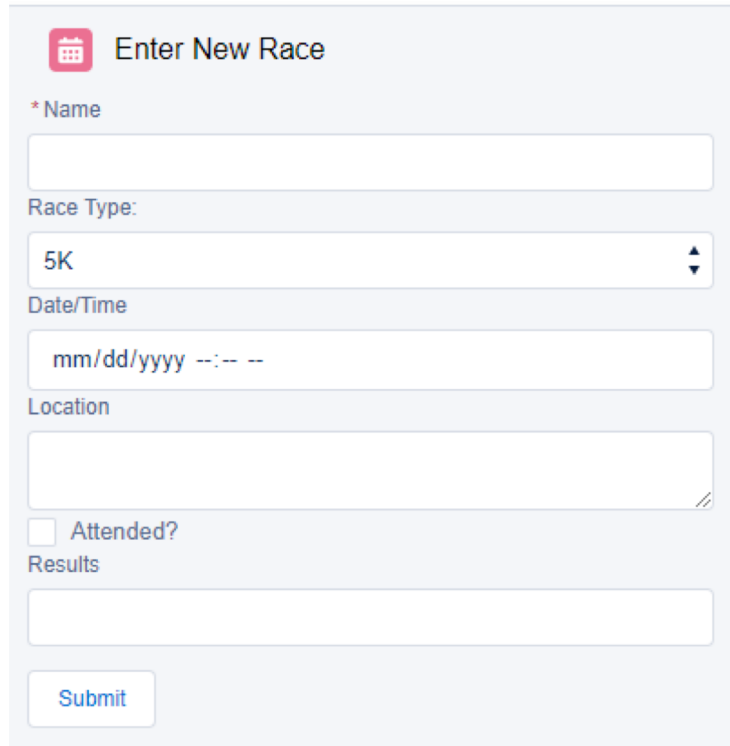
3 – Use the Lightning Data Service When Possible

AKA: The Standard Controller for Lightning



3 – Use the Lightning Data Service When Possible

AKA: The Standard Controller for Lightning



The screenshot shows a Lightning Data Service form titled "Enter New Race". It contains the following fields and controls:

- Name:** A required text field labeled "* Name".
- Race Type:** A dropdown menu currently showing "5K".
- Date/Time:** A date and time picker showing the format "mm/dd/yyyy --:-- --".
- Location:** A text area for location details.
- Attended?:** A checkbox.
- Results:** A text field for recording race results.
- Submit:** A button at the bottom of the form.

```
<force:recordData
  aura:id="raceRecordCreator"

  layoutType="FULL"

  targetRecord="{!v.newRaceRec}"

  targetFields="{!v.newRace}"

  targetError="{!v.errorMsg}" />
```


2 – Use Storable Actions When Possible

Caching in a Single Line of Code

ACCOUNT NAME	ACCOUNT NUMBER	ANNUAL REVENUE
Burlington Textiles Corp of America	CD656092	\$350,000,000
Dickenson plc	CC634267	\$50,000,000
Edge Communications	CD451796	\$139,000,000
Express Logistics and Transport	CC947211	\$950,000,000
GenePoint	CC978213	\$30,000,000

13 Accounts • page 1 of 3

Notice the Pagination

Elements	Console	Sources	Network	Performance	Memory	Application	»	⋮	×
top	Filter	Default levels							
Page 1 loaded in 545.1550000000002ms			<u>LightningDataGrid.js:38</u>						
Page 2 loaded in 297.7900000000009ms			<u>LightningDataGrid.js:38</u>						
Page 3 loaded in 271.0150000000067ms			<u>LightningDataGrid.js:38</u>						
Page 2 loaded in 0.6000000000058208ms			<u>LightningDataGrid.js:38</u>						
Page 1 loaded in 0.46999999998661224ms			<u>LightningDataGrid.js:38</u>						

2 – Use Storable Actions When Possible

Caching in a Single Line of Code

```
var action = component.get('c.getRacesDB');  
action.setStorable();  
  
action.setCallback(this, function(response) {  
    // Code here to handle the response  
})  
  
$A.enqueueAction(action);
```

ACCOUNT NAME	ACCOUNT NUMBER	ANNUAL REVENUE
Burlington Textiles Corp of America	CD656092	\$350,000,000
Dickenson plc	CC634267	\$50,000,000
Edge Communications	CD451796	\$139,000,000
Express Logistics and Transport	CC947211	\$950,000,000
GenePoint	CC978213	\$30,000,000

13 Accounts • page 1 of 3

1 – Use the Lightning Inspector Plug-in



The screenshot shows the 'Race Tracker Version 2' web application. The main form has fields for Name, Race Type (set to SK), Date/Time, Location, and Attended? Below the form is a 'Results' section. The Chrome DevTools component inspector is open at the bottom, displaying the component tree and props for the selected `<lightning-button>` component.

Race Tracker Version 2

Enter New Race

Name

Race Type

SK

Date/Time

Location

Attended?

Results

Component Tree

Performance Transactions Event Log Actions Storage

Expand All Show Global IDs Update On Select

```

<aura:if isTrue=function(cmp, fn) { return (cmp.get("v.page")=cmp.get("v.pages")); }> else=[ template=
[ ]>
    <lightning:buttonIcon onFocus=null onBlur=null iconName=utility:right variant=border size=medi
un disabled=false onclick={c.nextPage} type=button privateComputedButtonClass=slds-button slds-
button--icon-border privateIconClass=slds-button__icon ">
        <button class=private-button icon={iv.privateComputedButtonClass} ...>
            {ic.handleBlur} {iv.value} {ic.handleClick} {iv.accessKey} {iv.type} {iv.disabled} {ic.handleFocus} {ic.disabled} {iv.tabIndex} {ic.handleClick} {iv.privateTitle}>
                <lightning:primitiveIcon iconName={iv.iconName} variant=bare svgClass=
{iv.privateIconClass} privateSvgComputedClass=slds-button__icon ">
                    <span>
                        <aura:if isTrue=function(cmp, fn) { return ((fn.empty(cmp.get("v.alternativeText"))); }> else=
[ ] template=[ ]>
                            <aura:if isTrue=function(cmp, fn) { return ((fn.empty(cmp.get("v.footer"))); }> else=[ ] template=[ ]>
                                
```

Description: markup://aura/html

Global ID: 268:99:a

Is Rendered: true

Is Valid: true

HTML Elements: 4

Rerendered Count: 0

aura:id: privateButton

Attribute & Facet Value Provider

- <lightningbuttonIcon globalId="268:99:a"

onFocus=null onBlur=null

iconName="utility:right" variant="border"

size="medium" disabled="false"

onclick="{c.Paginator\$controller.nextPage}"

type="button" privateComputedButtonClass="slds-

button slds-button--icon-border"

privateIconClass="slds-button__icon">

Attributes

body: []

1 – Use the Lightning Inspector Plug-in

Component Tree Tab

Enter New Race

Name

Race Type:

5K

Date/Time

Elements

Console

Sources

Network

Performance

Memory

Application

Security

Audits

Lightning

Component Tree

Performance

Transactions

Event Log

Actions

Storage

Expand All

Show Global IDs

Update On Select

<lightning:select name="Type" value="{!v.newRace.Type}" variant="standard" disabled="false" readOnly="false" required="false" validity={}

onchange="null" onFocus="null" onBlur="null" privateComputedClass="slds-form-element" label="Race Type:">

class="{!v.privateComputedClass}" title="{!v.title}">

class="function(cmp, fn) { return fn.join(" ", "slds-form-element__label", (fn.eq(cmp.get("v.variant"), "label-hidden") ? "slds-m-right_xxx-small": ""); }" for="{!globalId}">

isTrue="{!v.required}" else=[] template=[]>

class="function(cmp, fn) { return

Descriptor: markup://lightning:select

Global ID: 29:0

Is Rendered: true

Is Valid: true

HTML Elements: 9

Rerendered Count: 4

aura:id: Type

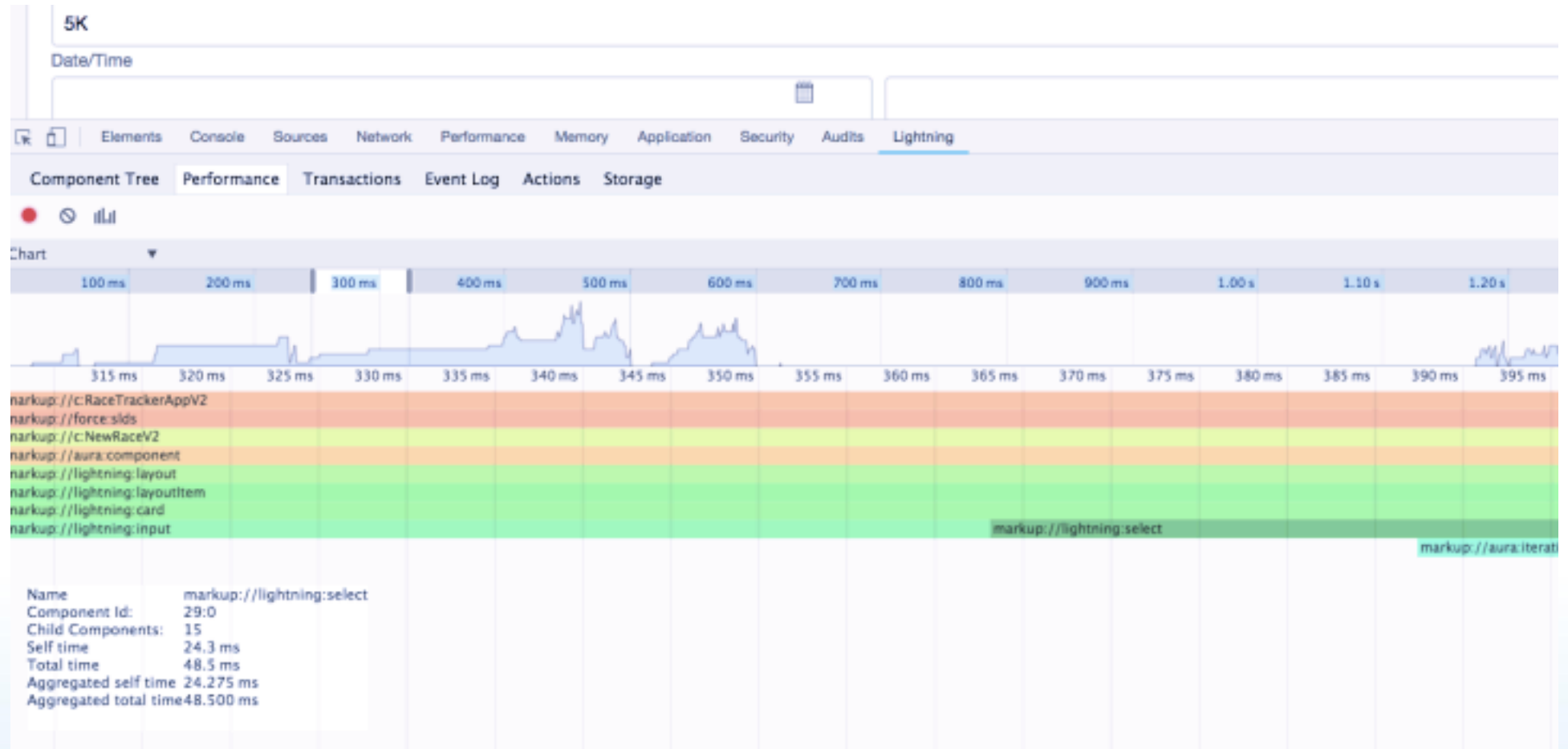
Attribute & Facet Value Provider

<c:NewRaceV2 globalId="9:0" newRace="{...}" raceTypes="[3]" isError="false" newRaceRec="{...}">

Attributes

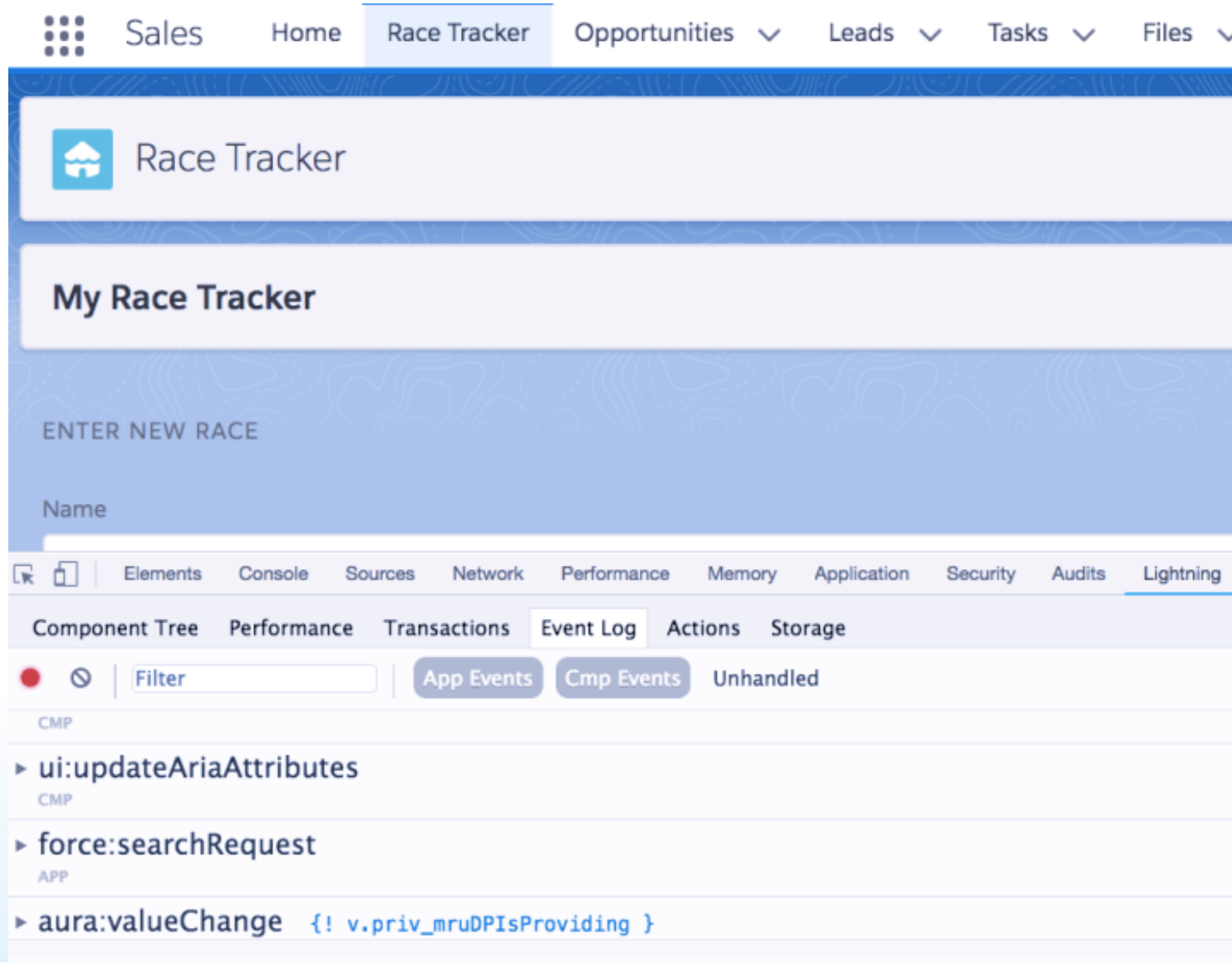
1 – Use the Lightning Inspector Plug-in

Performance Tab



1 – Use the Lightning Inspector Plug-in

Event Log Tab



The screenshot displays the Salesforce Lightning Inspector plug-in interface. At the top, the Salesforce navigation bar includes tabs for Sales, Home, Race Tracker (selected), Opportunities, Leads, Tasks, and Files. Below this, the 'Race Tracker' app header is visible, followed by a section titled 'My Race Tracker'. A large blue area contains the text 'ENTER NEW RACE' and a 'Name' input field. The bottom portion of the image shows the Lightning Inspector toolbar with tabs for Elements, Console, Sources, Network, Performance, Memory, Application, Security, Audits, and Lightning (selected). Within the Lightning tab, sub-tabs for Component Tree, Performance, Transactions, Event Log (selected), Actions, and Storage are present. The Event Log tab features a filter input, a 'Filter' button, and three event type filters: App Events (selected), Cmp Events, and Unhandled. A list of events is shown, including:

- CMP**
 - ▶ **ui:updateAriaAttributes**
CMP
 - ▶ **force:searchRequest**
APP
 - ▶ **aura:valueChange** `{! v.priv_mruDPISProviding }`

1 – Use the Lightning Inspector Plug-in

Event Log Tab



The screenshot displays the Salesforce Lightning Inspector plug-in interface. The top navigation bar includes tabs for Sales, Home, Race Tracker (selected), Opportunities, Leads, Tasks, Files, and Account. Below this, the 'Lightning' tab is active, showing a sub-menu with Component Tree, Performance, Transactions, Event Log (selected), Actions, and Storage. The Event Log tab is further divided into App Events, Cmp Events, and Unhandled. A 'Filter' input field is present. The main content area shows a 'Call Stack' table with two columns: 'Event Fired' and 'Handled By'. The first row shows an event fired from 'markup://c:AddToRaceList' with parameters '<c.NewRace globalId="706:0">' and handled by 'c.handleAddToRaces' with parameters '<c.ListRaces globalId="811:0">'. Below the table is a 'Toggle Grid' button. The grid view shows a sequence of four events connected by arrows: 1. (706:0) markup://c:AddToRaceList (black box), 2. (811:0) c.handleAddToRaces (red box), 3. (811:0) markup://aura:valueChange (blue box), and 4. (818:0) markup://aura:valueChange (blue box).

Event Fired	Handled By
markup://c:AddToRaceList <c.NewRace globalId="706:0">	c.handleAddToRaces <c.ListRaces globalId="811:0">

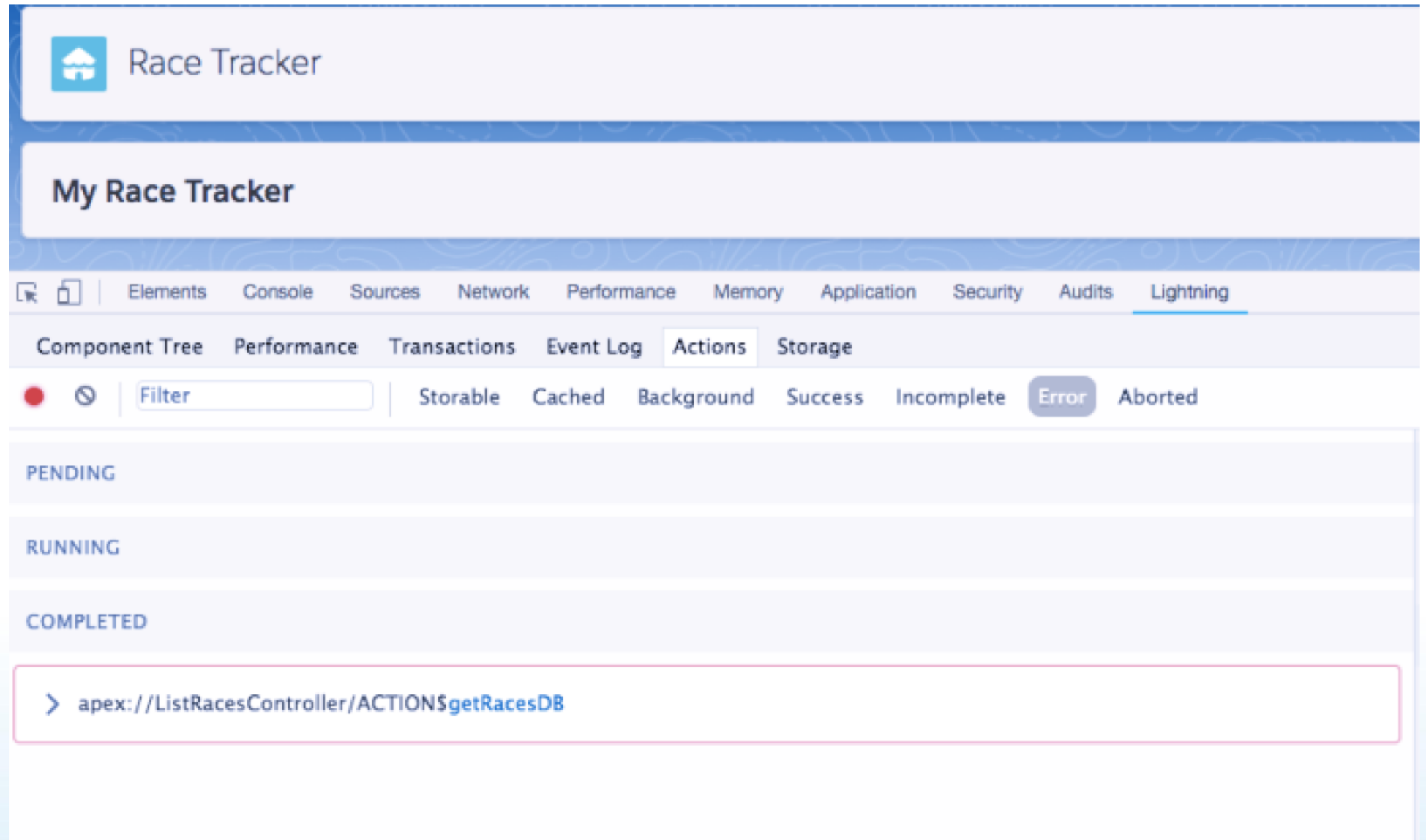
Toggle Grid

```
graph TD; A["(706:0) markup://c:AddToRaceList"] --> B["(811:0) c.handleAddToRaces"]; B --> C["(811:0) markup://aura:valueChange"]; C --> D["(818:0) markup://aura:valueChange"];
```

1 – Use the Lightning Inspector Plug-in



Actions Tab



The screenshot displays the Lightning Inspector plug-in interface within a web browser. The top section shows the application title "Race Tracker" with a house icon. Below it is a header "My Race Tracker". The main interface features a horizontal menu with tabs: Elements, Console, Sources, Network, Performance, Memory, Application, Security, Audits, and Lightning (which is selected). Under the Lightning tab, there is a sub-menu with "Component Tree", "Performance", "Transactions", "Event Log", "Actions" (selected), and "Storage". Below the sub-menu, there are filter options: a red circle icon, a filter input field, and buttons for "Storable", "Cached", "Background", "Success", "Incomplete", "Error" (highlighted), and "Aborted". The main content area is divided into three sections: "PENDING", "RUNNING", and "COMPLETED". The "COMPLETED" section contains a single entry: a right-pointing chevron followed by the text "apex://ListRacesController/ACTION\$getRacesDB".

1 – Use the Lightning Inspector Plug-in



Actions Tab

Component TreePerformanceTransactionsEvent LogActionsStorage

Filter

StorableCachedBackgroundSuccessIncompleteErrorAborted

PENDING

RUNNING

COMPLETED

▼ apex://ListRacesController/ACTION\$getRacesDB

Parameters

{}

Error

- [

- {

"message": "An internal server error has occurred\nError ID: 783526350-556574 (119852647)"

}

]

IdStateAbortableBackgroundCreated ComponentsStorable

425;aERRORfalsefalse0false

Component Source

<c:ListRaces globalId="816:0">

Demo Time



Top 6 Tips Review

6– Check Component Cache and Debug Settings

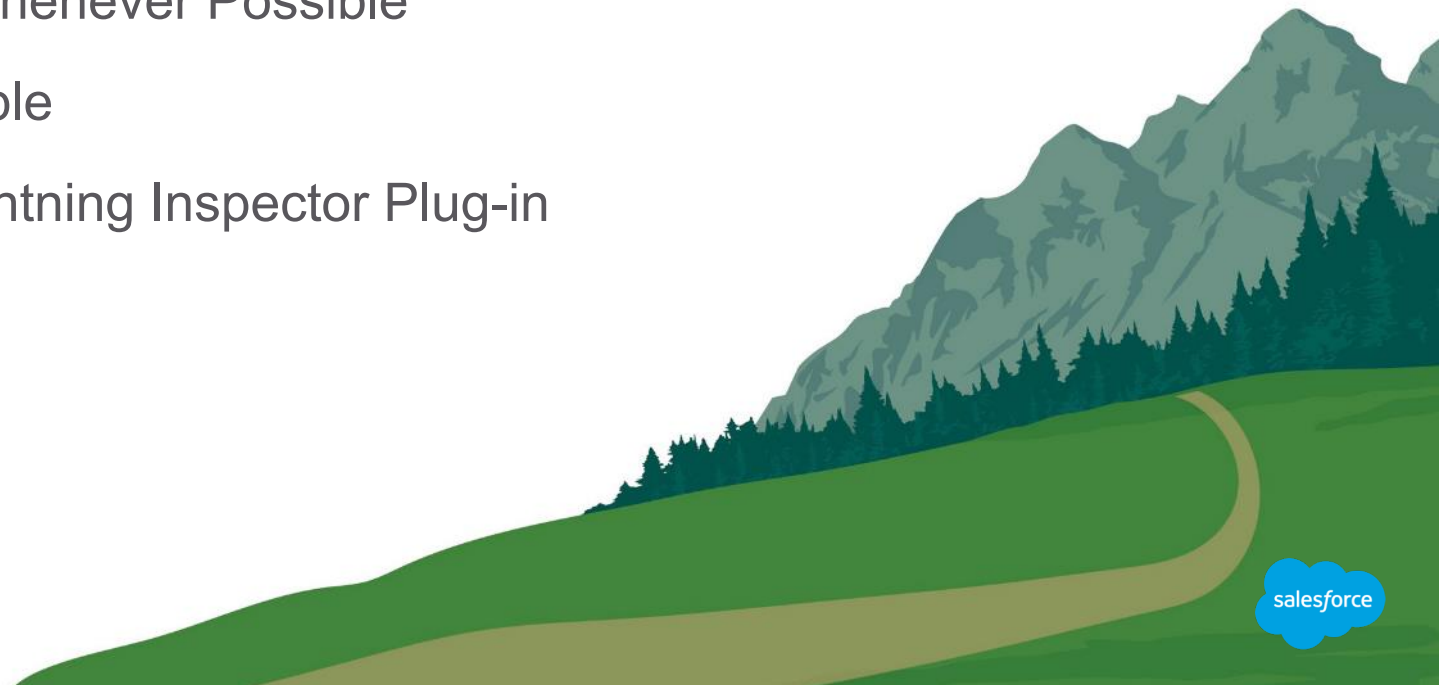
5 – Limit Use of Application Events

4 – Use <aura:if> for Conditional Rendering

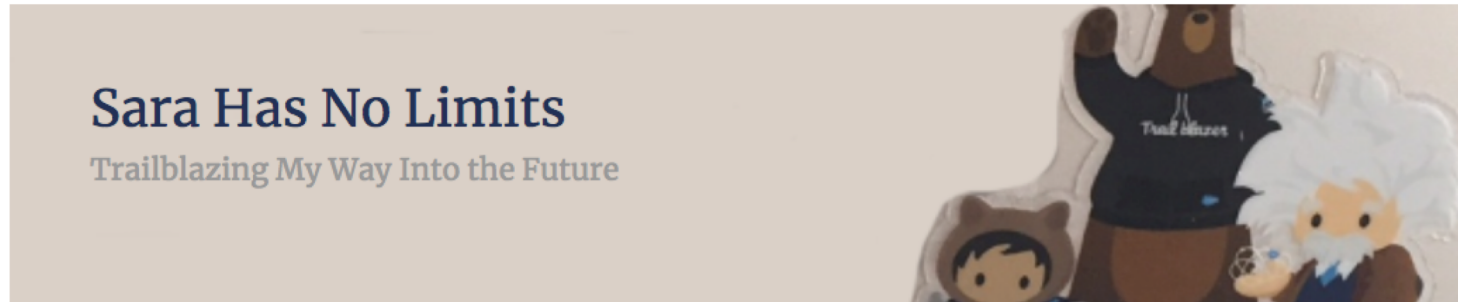
3 – Use the Lightning Data Service Whenever Possible

2 –Use Storable Actions When Possible

1 – Learn about your app with the Lightning Inspector Plug-in



Check Out My Blog For More Best Practices

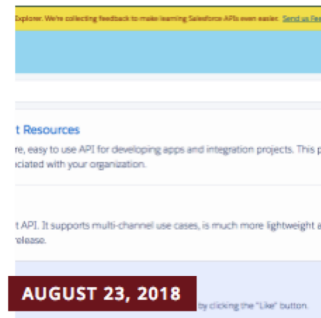


<http://saramorgan.net>

Home How I came to work for Salesforce...



Does someone mentioning Artificial Intelligence make



Check out the Salesforce API Explorer

New Pluralsight Course on Lightning

Lightning Component Development Best Practices

Follow Blog via Email

More Salesforce Resources

On Trailhead:

Lightning Data Service Basics

Lightning Component Core Concepts

Lightning Component Tips and Gotchas

On the Salesforce Developers Blog:

Lightning Inter-Component Communication Patterns

Lightning Components Best Practices

Caching Data with Storable Actions

Questions?

Reach me, Sara Morgan Nettles at @SaraHasNoLimits
or through my website at <http://saramorgan.net>



thank you

