Toast:toast are the message box or notification box that a developer can show according to the action.

```
import{ ShowToastEvent } from 'lightning/platformShowToastEvent';
```

Attributes:

1)Variant: Toast can of 4 types

a. **Info**
b. **Success**
c. **Warning**
d. **error**

2)title:

3)Message

4)Duration

5)Mode:

a)dismissible:toast will disappear either by the user or by the specified duration.

b)pester:toast will disappear after the specified duration.close button will not be there.

c)sticky:toast will disappear only after the user clicks the close button.

<u>Lightning Data Service</u>

:==> Predefined components plus more and more services on components can be implemented.

What is Lightning record Form

Lightning record form is easier than creating your own form manually.Lightning record form is used to quickly create a form to add,view or update a record.

AttributeS: Recordid,ObjectApiname & Fields[optional], layout,mode-readonlymode,edit,view

User can use view or edit mode both for display and input data.

Readonly for only view[only display]

Lightning:record form component provides you with the helpful features of

==>view

==>edit

==>display

Lightning:record form:no need for lightning input and outfield and all fields it will show

Lightning:recordedit[Edit]:Lightning input field required and output field only those fields which user has given.

Lightning:recordview[DISPLAY] Lightning input and lightning output field required.

```
<aura:component
implements="force:appHostable,flexipage:availableForAllPageTy
pes,flexipage:availableForRecordHome,force:hasRecordId,forceC
ommunity:availableForAllPageTypes,force:lightningQuickAction"
access="global" >
```

```
<!--<lightning:recordForm recordId="{!v.recordId}"
                    layoutType="Compact"
                    objectApiName="Lead"
                    mode="edit"></lightning:recordForm>-->
   <!--wHEN user pass recordid pon same page it is view and
editable both-->
   <lightning:recordForm recordId={recordId}
                    layoutType="Compact"
                    objectApiName="Lead"
                    mode="readonly"></lightning:recordForm>
  <!--This is only for read only-->

</aura:component>
```

…………………………………………………………………………………
…………..

```
<aura:component
implements="force:appHostable,flexipage:availableForAllPageTy
pes,flexipage:availableForRecordHome,force:hasRecordId,forceC
ommunity:availableForAllPageTypes,force:lightningQuickAction"
access="global" >
<lightning:card title="Account display">

   <lightning:recordEditForm recordId={recordId}
                        objectApiName={objectApiName}>
```

```
    <lightning:inputfield fieldName="Name"/>

    <lightning:button class="slds-m-top_small" type="submit"
label="Edit same record"/>

    </lightning:recordEditForm>


        <lightning:recordViewForm recordId={recordId}

                                  objectApiName="Account"

                                  >


            <lightning:outputfield
fieldName="Name"></lightning:outputfield>


        </lightning:recordViewForm>



 </lightning:card>



</aura:component>
```

lightning-record-form	lightning-record-view-form	lightning-record-edit-form

Create record forms to view, create, and edit records with very less coding.	Create customizable record display forms with read-only mode.	Create customizable record edit, view forms. You can also add some read-only fields.
Requires you to specify the `object-api-name` attribute	Requires you to specify the `object-api-name` attribute	requires you to specify the `object-api-name` attribute
The `record-id` is required in the edit mode	The `record-id` is required to load the record	The `record-id` is required to load an existing record.
No additional apex class required to load/create/update a record.	No additional apex class required to load a record.	No additional apex class required to load/create/update a record.
The `mode` attribute is required. `view` - used to load the record `edit` - used to edit/create the record `readonly` - record is opened in read-only mode	The record is always opened in the read-only mode.	The record is always opened in the edit mode.

The layout-type attribute is used to specify a `Full` or `Compact` layout. The fields configured by admin in the layouts are auto rendered on the form. `Full` - all fields configured on the full page layout of the record are rendered. `Compact` - all fields configured on the compact layout are rendered.	No fields attribute. Fields are rendered using `lightning-output-field` component nested inside the `lightning-record-view-form`	No fields atttribute. Fields are rendered using `lightning-input-field` component nested inside the `lightning-record-edit-form`
The `fields` attribute is used to mention a list of fields to be rendered.	-	-
The component expects the `fields` attribute or the `layout-type` attribute	-	-
Save and Cancel buttons are added automatically	-	Need to add a button with the submit to save the record. Also a cancel button is needed which can call the `reset()` function.

`lightning-record-form` does not support prepopulating of the fields when the form is loaded	-	Use the `value` attribute on `lightning-input-field` to prepopulate the field value.
If there are record types on the object, picklist values are loaded based on the record types. You must provide the `record-type-id` attribute if you have multiple record types and you don't have a default record type. Otherwise, the default record type id is used.	Not need as it's the read-only mode.	If there are record types, picklist values are loaded based on the record types. You must provide the `record-type-id` attribute if you have multiple record types and you don't have a default record type. Otherwise, the default record type id is used.
Use the `columns` attribute to define the number of columns	Use the `lightning-layout` and `lightning-layout-item` to create a multi-column layout	Use the `lightning-layout` and `lightning-layout-item` to create a multi column layout
You don't have the flexibility to add an empty space in the field grid between two fields.	You can add an empty space in the field grid using an empty `layout-item` between two fields	You can add an empty space in the field grid using an empty `layout-item` between two fields
You can't add custom elements	You can add custom elements to display fields	You can add custom elements to display/update fields

Custom Events • error • load • submit • success	Custom Events • load	Custom Events • error • load • submit • success
You can use the cancel button to reset the form.	-	To reset the form fields to their initial values, use the `reset()` method on the `lightning-input-field`.
Does not support custom UI level validations	-	You can add custom UI level validations on the `lightning-input-field` components.