**Creating Forms**

**Ask the right questions:**

**Explain what you want:**

Password requirement details

Show password for their verification

Clearly show the user mistakes in filling out the form / highlight

**Create a good flow:**

Makes your user comfortable (free / no credit card required)

Details to be collected in form should be in proper order

Get as minimum details as possible for a signup

Minimum questions to search a data and then you filter it up with other questions

Stressful parts in signup may move the users away, add it after signup

Give all the idea of the process (in check out, list all the process, click on the process list one by one to get the information)

Good confirmation is important (thank you, you can now edit question bank and benefit your surroundings, continue to QB)

**Data privacy:**

Some countries have certain laws on how to collect and maintain user details.

USA – childrens less than 13 should have parent consent to sign in social websites

Europe – only required data of users should be collected

Commercial mails should not be sent to users without their consent

Users should have the option of opting out the emails they receive

Should follow the laws of the countries at the receiving end of the mail.

**Creating Forms with HTML**

**Basic Form Structure:**

<h1>About you</h1>

**<form id="questions" method="POST" action="Htmlforms.html">**

<label **for="favcolor">**What is your favorite color?</label>

<input type="text" name="favcolor**" id="favcolor"**> // label for input with id favcolor

<input type="submit" name="submit" value="submit">

**</form>**

**Text, email, and URL input types:**

<input **type="text"** name="name" **placeholder=”enter name**” **maxlength=”25”><**br>

< input **type="email"** name="email" id="email" **multiple**><br>

input type url/email gives extra keyboard options in mobile

<input **type="url"** name="url" id="url"><br>

<input type="text" **pattern="[A-Za-z]{3}"> // regular expression ( {3} – not more than three letters )**

**Number Input type:**

<input **type="number"** name="number" id="number" **min="-10" max="10" step="2"**>

*step=”2” allows only even values and steps two places*

*Should not use input types for zip code as it will truncate initial 0*

**Date and time input types:** // not supported in all the browsers

**Date :** <input **type="date"** name="date" id="date" **min="2017-01-01" max="2017-12-31"**>

**Time :** <input **type="time"** name="time" id="time" **step="1800"> //***step in seconds*

**Telephone numbers:** <input **type="tel"** name="tel" id="tel">

*Tel type allows you to type any letters*

**Radio Buttons:**  radio / checkbox (just replace radio with checkbox for checkbox)

**<fieldset>**

**<legend>What is your favorite color?</legend>**

<input type="radio" name="favcolors" value="blue">blue<br>

<input type="radio" name="favcolors" value="red" checked>red<br>

<input type="radio" name="favcolors" value="green">green<br>

**</fieldset>**

**Select:**

**Multiple :** multiple selections

**Size :** box size of the options

**Optgroup :** categories for options

<select name="pizza" id="pizza" **multiple size="7"**>

**<optgroup label="veggies">**

<option **value="mushroom" selected**>mushroom</option>

<option value="onions">onions </option>

**</optgroup>**

**<optgroup label="meat">**

<option value="sausages">sausages</option>

<option value=" pepperoni "> pepperoni </option>

**</optgroup>**

</select>

**Multiple :** allows multiple selections

**Accept=”image/\*”** , only allows images files of all types to be uploaded .

Still virus or code files can be sent by user , it should be protected in the server side

**Text area and additional input types:**

**<textarea** name="poem" id="poem" rows="5" cols="60">**</textarea>**

<input **type="color"** name="color" id="color">

**<input type="file" name="resume" multiple accept="image/\*">**

<input **type="password"** name="password" id="password">

<input **type="hidden"** value="1234" name="tracking">

// hidden to the user and you can send any tracking detail along with user data

<input **type="range"** **min="0" max="100" step="2"> // just provides slider use it with js**

**<input type="search">** // users can enter search terms

**Submit and other types of buttons:**

<input **type="submit"** name="submit" value="submit">

<input **type="reset**" value="reset">

<input **type="button"** value="button"> // button but does nothing , use js

<**button type="sumbit">**place <strong > order</strong></button>

**<button type="reset">**reset</button>

**<button type="button">**button</button> // button but does nothing , use js

**<input type="image" src="" alt="bird" width="50px"> // image as button**

**Attributes used in forms:**

**required** -> the field should be filled

**pattern="[A-Za-z]{3}">** // regular expression ( {3} – not more than three letters )

**aria-required=”true”** // in type =”text” – backup for older browsers and screen readers

**disabled - >** no value can be added and no data is submitted

**readonly - >** no value can be added and but data is submitted

**Making forms attractive**

**Form submission and security:**

<form id="questions" method="POST" action=" ">

</form>

**Action:**

action=”example.php” - relative url

action=<https://www.example.com/page.php> - absolute url

sending and receiving page :

https -> https // everyting is ecrypted and secured

http -> http // data is not encrypted (no no for secured data)

http - > https // data is encrypted

https -> http // data is not encrypted and browser warns

**method:**

**get** : browser sends http request and asks for response with an empty body in the request

data in url – [https://www.example.com/**?color=blue&design=striped**](https://www.example.com/?color=blue&design=striped)

**post:** browser sends http request and asks for response with data in the body of request

Don’t rely on brower validation . hackers can send other datas or codes

**Form validation:** //telling users the mistakes while entering the data

Client-side validation ( on submit – validation by browser when form is submitted)

Client-side validation (inline – validation msg when filling out the form)

Server-side validation ( done by server when it sends back error messages)

Client-side : faster / not 100% secured . use combination of both client and server

***Highlight errors near the user input***

**input:valid{ // change the box color**

**border:2px solid rgb(53, 53, 212);**

**}**

**input:invalid{**

**border:2px dashed red;**

**}**

**Styling Form fields with CSS**

**Basic form styles: Text color and size:**

Form inputs may not inherit all the styles from the body , so

button,input,select,textarea {

**font-family:inherit;**

font-size:100%;

}

file button cannot be changed for security reason and it will have unusual results for css

selecting particular input : **input[type=”date”] {} || input[type=”button”]{}**

particular label : **input[type=”date”] + label // input date and just sibling/next label**

**Box sizing for forms:** // the elements may of different sizing so

input,select,textarea {

*-webkit-box-sizing: border-box; -webkit-box-sizing: border-box;*

**Box-sizing:border-box** //default: box-sizing:content-box

}

**Styling text inputs:**

**Width and padding for all inputs to look even**

**textarea{**

**vertical-align: middle;**

**}**

**Styling radio buttons and check boxes with images: // cannot change directly using their tag**

*Inside label :* **<label><input type="radio" name="gender" value="male">male</label>**

**label{**

**display:inline-block; //labels are inline and size of check box is not even so**

**}**

**To use different image for check box and radio button when selected**

**input[type='checkbox']{**

**opacity:0;** // dont use display:none , it will not display anything for users

**}**

**input[type="checkbox"] + label {**

**background: url(checkbox-empty.png) left center no-repeat;**

**background-size: 1em 1em;**

**padding-left: 1.5em;**

**margin-left: -1.5em;**

**}**

**input[type="checkbox"]:checked + label {**

**background: url(checkbox-checked.png) left center no-repeat;**

**background-size: 1em 1em;**

**}**

**Styling radio buttons and check boxes as buttons:**

<fieldset>

**<legend>gender</legend> // id and for enables label to be clicked to select radio button**

<input type="radio" name="gender" value="male" **id="male"**><label **for="male">**male</label>

<input type="radio" name="gender" value="female" id="female"><label for="female">female</label>

<input type="radio" name="gender" value="others" id="others"><label for="others">others</label>

</fieldset>

**input[type='radio'**]{

**opacity:0;**

width:0;

margin:0;

}

**input[type='radio'] + label{**

**display:inline-block;**

background-color:rgb(16, 41, 16);

color:rgb(243, 243, 243);

border:2px solid rgb(173, 64, 64);

width:4em;

text-align:center;

border-radius:7px;

}

**input[type='radio']:checked + label{**

background-color:rgb(104, 90, 224);

color:white;

font-weight: bold;

}

**Styling select:**

select {

display: block;

margin-bottom: 20px;

-webkit-appearance: none;

-moz-appearance: none;

**appearance: none;**

background-color: white;

}

**select:not([multiple]) { // select element that does not have multiple prop**

background: url(double-arrow.png) right center no-repeat;

background-size: 1em 1em;

padding-right: 1em;

background-color: #fdebd0;

}

select {

padding: 8px 1em 8px 8px;

background-color: #fdebd0;

border: 2px solid #af601a;

}

option {

padding: 5px;

}

optgroup {

border: 1px dashed #af601a;

padding: 5px;

}

**Styling buttons: // search pnline for button generator**

If you change the button styles (even-background color) the default styles disappear.

**#styled input[type="submit"]:focus** {

border: 2px solid green;

outline:none;

}

#styled input[type="submit"]**:hover** {

border: 2px solid red;

}

#styled input[type="submit"]**:active** {

background-color: orange;

}

**Styling pseudo-classes: pseudo-class->specifies a state of an element (hover,active etc)**

input:**focus**{

background-color:green;

}

input:**active{**

background-color:red;

}

**:disabled**{

background-color:gray;

}

**:enabled**{

background-color:blue;

}

input[type="checkbox"]**:checked** + label{

background-color:blue;

}

select**:selected**{

background-color:green;

}

input[type="number"]**:valid**{

border:2px solid green;

}

input[type="number"]**:invalid**{

border:2px solid red;

}

input[type="number"]**:in-range**{ // for range and pattern

border:2px solid green;

}

input[type="number"]**:out-of-range**{ // for range

border:2px solid red;

}

input**:required**{

border:2px solid green;

}

input**:optional**{

border:2px solid green;

}

**Form layout with CSS**

**Small-screen layout for forms:** // screen size 320px -> in chrome inspect move the screen to see size

label{

display:block; // as input and label should be in next line (320px)

}

input[type="checkbox"] + label{ // selector - > adjacent sibling

display:inline; // checkbox options in single line

}

fieldset{

border-width:0; // taking out the border;

}

Input[type=”text”],input[type=”number”],input[type=”email”]{

Width:90%; // making these inputs to align to the screen well

}

**Wide-screen layout for forms:**

**http://howbigismybrowser.com/=> to know the current screen size**

Use media queries **@media(min-width:520px){}**

**Form labels:**

**Placeholder attributes:**

For small forms use place holder but for forms with many details try not to use.