

# Best Practices For “Internationalization of E-Governance applications For India”



## **Executive summary:**

It's a simple question to ask when you build applications: Who is the user?

If most users of applications in India are Indians, then there are many preferences unique to our country and cultural practices. The primary one is that most of us don't speak English.

This question drives everything in this document. The outcome is usable applications that deliver usefulness and thereby increase the user-base of applications (because people who don't speak English can use them). In this sense it applies to all applications in general and eGovernance applications in particular.

The method to do this is also simple: Treat all Indian requirements as if they are in English and find their equivalent Indian resources.

The outcome of this approach is applications that are simple, seamless and complete for every user in India.

If the primary measure of the success of an application is the number of people who use it, then Internationalized/localized applications well is guaranteed to increase the success of applications.

## **Measurements (What to look for)**

Take any arbitrary application. Ask yourself who the user is. Ask if that person can use the functions offered in the same way in the different languages of the targeted area. If it can then, the application is well Internationalized / Indianized.

The key features to look for are:

- i. Can all the services available be offered in the relevant languages?
- ii. Is there the means to handle the requests generated by users in these languages?
- iii. Can a transaction be completed in an Indian Language?

If the answer to these questions is "yes", then it demonstrates that an arbitrary user who does not speak English can, (possibly with a little help from someone who knows how to operate a computer / kiosk) consume the services offered in a manner that meets the objective.

## Outcome

There have been 5 stages in the development of the internet.

1. Publish: Information is published for reading. There is no ability to respond.
2. Interact: Apart from publishing, users can interact with each other, by publishing on pages and getting responses from the providers.
3. Transact: Perform a commercial transaction of some value such as a bank transaction or purchase.
4. Integrate: This was when large entities were able to build on the previous 3 stages and connect different wings of their organizations to perform transactions. This dramatically speeded up the speed of activities. The purchase of railway tickets online in India is a good example.
5. Seamless: This is when all activities were integrated and led to dramatic advantages to organizations of all kinds.

**This is the age we live in.**

It is our objective now to demonstrate everything stated above available to all Indians regardless of their knowledge of English. It will be native to users' experience and expectations.

The underlying infrastructure is ready and the implementation can be done. The outcome is cost-effective delivery of useful services to all Indians using the internet.

The mantra is: **Simple, Open, Everywhere.**

The rest of the document is about how to accomplish these outcomes.

## Structure of this document

Abstract (above)

Measurements (above)

Outcome (above).

## How to accomplish the outcome

### Overview

There are two rules to accomplish seamless Indian applications:

- i. Design for Internationalization/Indianization and
- ii. Use standards.

### Designing for Internationalization/ Indianization:

Design is at the core of the application. You should plan for the user and not the developer or administrator. The practices for Internationalization / Indianization are well known and are summarized below:

- i. Single application that is adapted to users.  
The functional aspects of the application should be mapped to users. It should be designed to include **all** aspects of user choice that are different. This is best accomplished by breaking user-preferences up into parameters and then allowing users / applications to select them depending upon the user.
- ii. Separation of code and data.  
Parameterization of user-preferences is not possible if code and data are not separate. This separation should be done before a single line of code is written. This includes application-related code like labels for interfaces in which data is entered and business-logic for processing user-generated requests.  
If this is accomplished then it doesn't matter what languages the data is in, it can be processed and transacted.
- iii. Choosing code elements/artefacts that can be replaced or upgraded to include the preferences or rules that apply to different users.  
This does not necessarily mean that the business logic of applications cannot accommodate geographical differences. But it would mean that they can be invoked/called up according to the applicable rules.

- iv. Tying up loose ends: This is the hardest aspect of applications-design. Unless the application pre-assess and accommodates user expectation from the beginning, there will always be rough edges that will show. Seamlessness cannot be achieved without proper requirements assessment/gathering that is done before a single line of code is written.

## Using standards

These are:

- i. Unicode: Without it, no other standards can be applied.
- ii. W3C: Includes Unicode and all other standards for interoperability.
- iii. INSCRIPT: For Input. This is India-specific and simplifies keyboard-based entry.

If these 3 standards are used, it is guaranteed that the end result is “perfect” application interoperability that includes Internationalization

There may be other standards that apply for the organization for which the application is intended. But adherences to these 3 standards are adequate to ensure that there are no platforms or inconsistencies to supporting the application across platforms, devices or domains.

## The relevant platforms considered are:

- i. Clients:**
  - a. Windows
  - b. Linux
  - c. Macintosh
  - d. (Mobile interfaces separately below)
- ii. Operating Systems:**
  - a. Windows
  - b. Linux
  - c. Unix flavours
  - d. Mainframes
  - e. Virtualized OS and platforms.
  - f. (Mobile OS for mobile devices)

- iii. Middleware:**
  - a. Databases
  - b. Application-servers.
  
- iv. Web**
  - a. Browsers and browser-dependent clients.
  - b. Mobile browsers.
  
- v. Cloud and cloud platforms.**
  
- vi. Mobile/Handheld platforms**
  - a. Windows OS
  - b. Android
  - c. iOS
  - d. Blackberry
  - e. Symbian
  - f. Linux

Where there is support lacking (primarily in mobile platforms), they will become available as the usage grows.

The combination of these two approaches will deliver applications to users the same way in which English speaking users have come to expect applications to work.

In the rest of the document, we expand on these points and give a functioning guide to building and maintaining Internationalized / Indianized applications.

## Table of Contents

1. Introduction
  - 1.1 Background
  - 1.2 Who should use this document?
  - 1.3 Need of web standards and specification in Indian Language
2. Characteristics of Internationalized/Indianized applications
3. Designing For Internationalized/Indianized Application
  - 3.1 Internationalization/Indianization
  - 3.2 Localization
  - 3.3 Challenges in Software Globalisation
4. Components of Localization
  - 4.1. Unicode
    - 4.1.1 What is Unicode?
    - 4.1.2 Principles of Unicode Standard
    - 4.1.3 Character Encoding code pages
  - 4.2. Locale Support
    - 4.2.1 What is Locale Data?
    - 4.2.2 Common Locale Data Repository
  - 4.3. Keyboard and Display
    - 4.3.1 OTF (Open Type Font)
    - 4.3.2 WOFF
    - 4.3.3 Inscript Keyboard
    - 4.3.4 Phonetic Keyboard
    - 4.3.5 Statutory Requirement for New Rupee Symbol

## 5. Code Elements

5.1 Language identification and Negotiation BCP 47/RFC 5646

5.2 ISO 639:1, 2, 3, 5-Language Tag Standard

5.3 CSS

5.4 XML

5.5 XHTML

5.6 X-Forms

5.7 HTML 5.0

5.8 Web Accessibility

5.9 E-governance as Web Services

5.10 Mobile Web and its implementation

## 6. E-Governance Issues

## 7. Evaluation

7.1 Layout and design consistency

7.2 Information that support all browser

7.3 Accessibility Architecture

## 8. Use Cases

## 9. Need of W3C Validation services for Indian websites

9.1 CSS Validator

9.2 XHTML Validator

9.3 Mobile OK Validator

## 10. Important W3C Standards to be adopted

## 1. Introduction:

### 1.1 Background Information:

E-governance is more than just a government website on the Internet. But what is it exactly? What are the benefits of e-governance? What can governments do to make it work?

Solutions to development issues often require changes to government processes, e.g. by decentralisation. Objectives are generally to improve efficiency and effectiveness and to save costs. The driving force can also be public demand for online services and information that increase democratic participation, accountability, transparency, and the quality and speed of services. The implementation and use of ICT solutions can support governance reforms.

E-governance will become more and more present around the world in the next few years. Internationally most countries are in the early stages of e-governance. A good start has been made in Europe, USA and in other Westernised countries such as Australia and Singapore. Over the coming years also developing countries and their citizens can also benefit from e-governance.

This report explains what is meant by e-governance. It starts with a definition of e-governance; then presents a general e-governance model and several case studies and examples. Technology aspects are discussed, followed by a SWOT analysis on e-governance in developing countries. Finally, a description is given of what steps have to be taken to set up a policy on e-governance and how implementation projects can be selected.

Since the mid-1990s, UNDP has supported the implementation of ICT and e-governance programmes. However, in the last 5 years the field of e-governance has experienced dramatic changes perhaps best reflected by a move from technology driven initiatives to citizen-centred approaches and programmes, complemented by efforts to link the area to the broader theme of democratic governance.

In 2005, UNDP organized its first global Cop in Dakar, Senegal which focused on the completion of a UNDP Practice Note on e-governance. Three years later there is clear need to revisit the field of e-governance from a UNDP perspective to assess current programming and provide concrete ideas on future developments. It is indeed time to take stock to assess the impact e-governance has had in terms of enhancing democratic governance in order to strategically improve UNDP's and partners interventions towards improving the effectiveness of public service and information delivery, increased stakeholder involvement in decision-making processes, and the accountability and transparency of government (national, regional and local) processes.

In addition, although large amounts of public funds are going into e-governance (most being directed to e-administration or public investment inside government institutions), very few e-governance strategies, policies and programmes directly target those people with most need of services – the poor and vulnerable. This is yet another essential opportunities which remains to be harnessed systematically and which clearly links to the MDGs.

Access to information is also a fundamental issue that needs to be tackled in the context of e-governance. Fostering people participation in public policy making - a key aspect for truly fostering democratic governance processes- is essential to achieve sustained human development in any given country. Freedom of information acts and related legislation are certainly a first step. But opportunities also exist for people and citizens being capable of producing, summarizing, packaging and distributing public information that would otherwise never reach the vast majority of the population.

A recent mapping of e-governance projects for 2007 indicates that UNDP is supporting over 100 programmes in 51 countries with a total investment of almost 102 million USD. These projects range from policies and strategies to e-participation and access to information via ICTs. Clear links to both Public Administration Reform (PAR), local governance and anti-corruption also exist. Despite a considerable number of projects, little effort has been put into assessing the impact and lessons learned in terms of development impact on poor people, and methodologies for engaging communities in info-mobilization.

UNDP's more recent approach to e-Governance and Access to Information (via ICTs) has been an opportunity to provide support to governments and civil society organizations in using ICT to deliver better public services and information, and enhance the participation and involvement of citizens through networking in the various governance-related processes.

## **1.2 Who should use this document?**

This document provides guidance for developers of e-governance that enables support for national deployment. Enabling national deployment is the responsibility of all content authors and is relevant from the very start of development. Ignoring the advice in this document, or relegating it to a later phase in the development process, will only add unnecessary costs and resource issues at a later date.

It is assumed that readers of this document are proficient in developing e-governance application this document limits itself to providing advice specifically related to e-government.

## **1.3 Need of web standards and specification in Indian Languages.**

Standards give us lots of things:

- They allow us to write web pages that display on most browsers.
- They allow us to write XML applications that will work in other locations than where we wrote them.
- They allow us to switch between computers and software without difficulty.

## 2. Characteristics of Internationalized/Indianized Application

- i. Single executable:**
  - a. One application that is adapted for different languages. This makes it easy to manage consistently for any number of languages.
- ii. (Indianization) / Internationalization:** Designed for India. This means that all aspects of the expectations and cultural conventions of an Indian user is included. This can be extended to any other country in the world.
- iii. Localization:** Populated with India. This means that the elements of the application are translated and adapted for Indian users. This too can be extended for any country in the world.
- iv. Continuity and Consistency:** This means that the application is recognized as one that is “for the user” it is not in English or in any other language but native to the user’s language.
- v. Integrated:** All functions that are available to English speaking users are also available to non-English speakers.

## 3. Designing for Internationalized/Indianized Application

### 3.1 Internationalization/Indianization

Internationalisation is the Design and development of a product, application or document content that enables easy localization for target audiences that vary in culture, region, or language.

#### Why Internationalisation/Indianization:-

Internationalization significantly affects the ease of the product's localization. Retrofitting a linguistically- and culturally-centered deliverable for a global market is obviously much more difficult and time-consuming than designing a deliverable with the intent of presenting it globally. (Think back to the Y2K effort and trying to "undo" two-character year fields that were built on the assumption of "19xx").

So ideally, internationalization occurs as a fundamental step in the design and development process, rather than as an afterthought that can often involve awkward and expensive re-engineering.

### 3.2 Localization

Localization is the process of adapting of a product, application or document content to meet the language, cultural and other requirements of a specific target market. The purpose is to deliver the same set of functionalities and features with a different human language interface. Localisation involves not only translation, but additional customisation including numbers, dates, times, currency, sorting, icons, colours, etc.

## Why Localization?

Software localization is beneficial for both developers and customers. We should take into account the fact that the number of non-English speakers that use localized software is in a continuous growth.

Most software users expect that their software be written in their own language. Even if localization is done at a certain cost for software companies, it entails obvious benefits: the users that perfectly understand a product can manage it more properly, can use it more efficiently, and are less prone to making costly mistakes.

All these have an influence on the final result. More competent and efficient users translate into fewer costs for support and customer service.

Localization allows users to interact with software in their own language via an intuitive configuration for them.

### To exemplify:

- Messages are in their native language.
- The entry fields are formatted in a manner that is common to their respective country (name, address, date, hour, etc.)
- Different types of keyboard are taken into account for entering information.
- The error messages in the native language guarantee quick solutions for the problems.

## 3.3 Challenges in Software Globalisation

There are several kinds of issues that arise in the process of releasing software into a new locale, and most of the times these issues requires the programmers to change the source code to handle the scenarios specific to different communities.

A completely internationalised product is never supposed to undergo a change in source code during the localisation process.

Some common problems faced during software localisation are discussed below.

## **Text Input / Output**

Often software is unable to handle the characters of the target UI languages because during development it was designed to address a single human language. So it might fail to handle the characters of target UI languages and display junk characters instead of proper ones.

The following errors are generally observed in the area of Text I/O.

- Costs of Localization and Impact on Pricing.
- Language and Content.
- Currency.
- Time and Date Formatting.
- Measurements.
- Formats (Paper and Envelope Sizes, Address Formats).
- Collating Sequences.
- Numeric Formats.
- Color Scheme.
- Images and Sounds.

## **Text editing and fonts:**

In text editing the general problems can be isolated to the following areas.

- Keyboard shortcuts might need to be changed according to the translated text that contains these shortcuts.
- The font name, size, and style in dialog boxes and documents can be changed to customize the final UI based on user preference.
- Incorrect cursor positioning.

- Inconsistent click, double-click and multi-click selections

### **Locale specific knowledge:**

Locale-aware software means that the functional behaviour of software and display of UI elements and text varies as per the rules of a particular locale. In general this covers the following issues.

### **Number formatting:**

Consider the examples below to understand the variations present in the number formatting in different locales.

- 12,34,56,789.00 in India (One grouping on thousand and then grouping on hundreds)

### **Currency formatting:** \_currency symbols

- ` 4,567.89 (Indian Rupee)

### **Date formatting:**

Elements like order, width of fields, separators, padding with zeros etc.

- DD-MM-YY (India Date)

### **Time formatting:** It is also governed by locale.

- 10:00:15 PM (India Time)

**Non-Gregorian calendars:** In different countries different calendars are used, even in a single country there can multiple calendars used. Some common calendars in use: Gregorian, Arabic Lunar, Buddhist, Japanese and so on.

**Measurement units:** There will be problems if the software does not handle the unit measurement system of that locale properly or does not provide a mechanism for inter-conversion, common systems in use are foot, pound, second (FPS) and meter, kilogram, second (MKS).

**Address formatting:** Software that deals with postal addresses will show error messages or will display the wrong output if it is not designed to handle the address formatting as per the selected locale of the software.

**Personal titles:** One needs to be careful in deciding the length of fields for the names/family names/various titles .Ordering of title, first name, family name is also important and varies from locale to locale.

- Generational suffix (Jr., Sr., II, III)
- Honorific (Mr., Mrs., Ms., Miss, Dr., Master)
- Title (President, Director)
- Rank (Lt., Maj., Col.)
- First name
- Family name

**Telephone number formatting:** The problems in this section are generally caused by mishandling of the:

- Number of digits
- Digit groupings (country code, area code, local number)
- Digit separators (spaces, hyphens, periods, parentheses)
- Use of extensions

### Searching and sorting:

The sorting order of data and the ability to search are among the most critical and important features of the software. The software may sort incorrectly and not find the searched text if the following issues are mishandled in a particular locale.

- Binary order  $A < C < Z < a < c < z < Ç$
- Code Point Order (same as UTF-8 binary comparison)
- UTF-16 Order (Java String binary comparison)
- Refinements, usually only for matching, not sorting
- Case-insensitive
- Matching equivalent forms of text

## UI related issues:

Translated strings can be of different height, width and length so the UI elements' size for the source strings may not be suitable for the target language. This can cause truncation of the UI elements.

Additionally we may face problems related to duplicate and non-functioning hotkeys and keyboard shortcuts.

## 4.Components of Localization

- i. Unicode: Without Unicode, it is impossible to offer users seamless localization and total interoperability across platforms. It is the cornerstone for computing just like ASCII. You cannot have an Indian application without Unicode.
- ii. Components using Unicode for the application :
  - a. Fonts for display of Unicode characters.
  - b. Keyboards for entry of characters.
  - c. W3C / Internet tags for correct layout and usage of text. (Including CLDR here.)
  - d. Other tools for application-enhancement and completeness: Like text validators and spell checkers.

### 4.1 Unicode

Unicode is like ASCII. It is the only basis of internationalization in the world. All world standards that enable applications to be designed for localization (Indianization) depend upon it.

#### 4.1.1 What is Unicode?

Unicode is the standard for information interchange worldwide as all IT Companies support it. It is the Universal character encoding standard, used for

representing text for information processing. Unicode encodes all of the individual characters used for all the written languages of the world. The standards provide information about the character and their use. Unicode uses a 16 bit encoding that provides code point for more than 65000 characters (65536). It assigns each character a unique hexadecimal numeric value and name. Unicode and ISO10646 Standard provide an extension mechanism called UTF-16 that allows for encoding as many as a million. Presently Unicode Standard provides codes for 49194 characters.

#### 4.1.2 Principles of Unicode Standard

- i. **Universality:** It applies to all the scripts of the world. This means that all Indian scripts are addressed.
- ii. **Efficiency:** All characters are equally accessible and there are no modified states in the encoding. This means all characters are pure and have no enhancements.
- iii. **Characters, not glyphs:** The visual characters are not represented (except mistakes) but only the component text elements.
- iv. **Semantics:** These are used to enhance item “efficiency” above to allow components to be used correctly in the input/output/processing context so that the user experience is perfect.
- v. **Plain text:** Unicode characters when used in isolation represent only the characters themselves. These can then be enhanced with tags and other markers in the context of their use to get the intended results.
- vi. **Logical order:** This is how Unicode characters are stored in memory devices. It is usually left to right. Logical order needs to be converted to the required physical/processing order to get the required result. For example, Unicode characters for Urdu could be stored from left to right (in logical order) but would be required to be converted to their physical order (right to left) to be read correctly.
- vii. **Unification:** There is no duplication in how Unicode characters are encoded. This is made possible by the efficiency rule.
- viii. **Dynamic Composition:** Unicode allows for component characters to be combined with others to form other characters.

This is specially applicable to Indian Languages where most “matras” are formed using this.

- ix. **Stability:** Most importantly, this means that once Unicode characters are assigned, their code point assignments cannot be changed, nor can characters be removed. Characters are retained in the standard, so that previously conforming data stay conformant in future versions of the standard. Sometimes characters are deprecated
- x. **Accurate Convertibility:** Because of principle IV above, Unicode data is fully convertible with any unique mapping required for other standards. This means that there is guaranteed one-to-one mapping between Unicode and ISCII. This makes all kinds of migration possible.

### 4.1.3 Character Encoding code Pages

There are many ways to represent Unicode code points. These are called Unicode Transformation Formats. For the purposes of this document and general purpose applications, UTF-8 is the preferred format of encoding.

For more details : [www.unicode.org](http://www.unicode.org)

## 4.2 Locale Support

The Unicode CLDR provides key building blocks for software to support the world's languages, with the largest and most extensive standard repository of locale data available. This data is used by a wide spectrum of companies for their software internationalization and localization, adapting software to the conventions of different languages for such common software tasks as:

- formatting of dates, times, and time zones,
- formatting numbers and currency values
- sorting text
- choosing languages or countries by name

Language matching is used to find the best supported locale ID given a requested list of languages. The requested list could come from different sources, such as such as the user's list of preferred languages in the OS Settings, or from a

browser Accept-Language list. For example, if my native tongue is Hindi, I can understand Punjabi and Tamil, my Bangla is rusty but usable, ideally an implementation would allow me to select preferred list of languages, skipping Bangla because my comprehension is not good enough for arbitrary content.

Language Matching can also be used to get fallback data elements. In many cases, there may not be full data for a particular locale. When such fallback is used for resource item lookup, the normal order of inheritance is used for resource item lookup, except that before using any data from root, the data for the fallback locales would be used if available. Language matching does not interact with the fallback of resources within the locale-parent chain. For example, suppose that we are looking for the value for a particular path P in nb-NO. In the absence of aliases, normally the following lookup is used.

The language Matching data is interpreted as an ordered list. To find the match between any two languages, use the likely sub tags to maximize each language, and perform the following steps.

- Remove any trailing fields that are the same.
- Traverse the list until a match is found. (If the one way flag is false, then the match is symmetric.)
- Record the match value.
- Remove the final field from each, and if any fields are left, repeat these steps.

#### 4.2.1 What is Locale Data?

A locale represents information on the rules and preferences that is specific to the end user's particular country, language and territory. For example, the holidays of India are a part of its locale.

The data associated with locale provides support for: presenting, formatting, parsing of local data elements like dates, timestamps, numbers, currencies, measurement units, sort-order (collation), holidays, calendars, translated names for time zones, languages, countries and Scripts.

These elements can be used (and ultimately enhanced) to present a complete set of “user-environment variables” for the application to be presented to the user as completely native.

When used properly, every element of a user’s application-use environment is according to his expectations, practices and conventions. It allows users to work completely according to his real-world environment.

#### **4.2.2 Common Locale Data Repository**

CLDR the largest standard repository of locale data in the world. It is a part of the W3C and Unicode Standard. It provides locale data in an XML format for use in computer applications. It facilitates locale-related information sharing among applications regardless of their domains. Its goal is to provide basic linguistic information for diverse “locales” in an open, interoperable form.

This data is usable for localizing applications.

Some examples of the information that CLDR gathers for languages and territories are:

1. Date formats
2. Time Zones
3. Number formats
4. Currency and its formats
5. Measurement Systems
6. Collation (Sort order) Specification: Sorting, Searching and Matching
7. Translations of names for language, territory, script, time zones, currencies
8. Script and exemplar characters used by a language.
9. Calendaring rules, Formats and important dates.
10. Specification of selected but universal cultural terminologies.

There are more types of data and it is being enhanced all the time to increase precision. It is the standard source of locale data worldwide.

For more details: <http://cldr.unicode.org/>

### 4.3 Keyboard and Display (Output and Input)

The use of Unicode addresses all requirements in applications. This section briefly describes how Input and Output works. Output is addressed first.

#### 4.3.1 OTF (Open Type Font)

Open Type fonts convert the Unicode code numbers to their glyphs on the display interface. They are directly based on Unicode. Open Type provides a series of enhancements to the TrueType format, the most significant of which allows PostScript font data to nest inside a TrueType software “wrapper.”

##### **The main capabilities of Open Type are:**

- Broadest multiplatform support
- Enabling fonts to address large character sets.
- Improved publishing for the internet.
- Better protection for fonts against piracy and duplication

##### **Multiplatform Support:**

Open Type is a superset of the existing TrueType and Type 1 formats, and provides support for both type in print and on-screen. Open Type fonts works right “out of the box.” Both PostScript and TrueType versions of Open Type are supported across all platforms, making fonts easier to use and more versatile.

Your application needs to use Unicode and the UTF-8 encoding. The user’s platform needs to have an Open Type font available on the Operating System for it to be used by the browser.

##### **Large Character Sets:**

Open Type allows type designers and font foundries to create larger character sets within fonts. Within the parameters of the TrueType and Type 1 formats, fonts are limited to 256 characters. If a typeface designer wanted to create an extended ligature set, small caps, swash and alternate characters, or characters to support multiple languages, these

had to be put into another font. The large character set capabilities of Open Type allows type designers much more latitude in typeface design, resulting in better graphic communication.

This enables Indian Languages to be addressed “perfectly” by Open Type fonts and all characters can be shown as they are supposed to.

### **Improved Internet and PDF Publishing:**

Open Type makes it possible for Web page creators to include high quality on–screen fonts with their online documents. This means that designers are able to produce typographically richer documents and reduce the time required to download and display these documents on screen.

### **Better Font Protection:**

Open Type fonts also contain a “digital signature” that allows operating systems and browsing applications to identify the source and integrity of fonts – including embedded font files obtained in Web documents. Font developers are able to encode embedding restrictions in Open Type fonts to maintain better control over how their fonts are used.

### **Font Embedding:**

The Open Type format allows for font embedding. This means that a font can be included with a file and sent to someone else. There are four approaches to Open Type font embedding:

- Font embedding that allows the document to be viewed on screen and printed
- Font embedding that allows viewing, printing, and document editing
- Font embedding that allows viewing, printing, editing, and installing onto a hard drive
- No embedding allowed.

#### **4.3.2 WOFF (Web Open Font Format)**

This format was designed to provide lightweight, easy-to-implement compression of the font data, suitable for use in conjunction with the @font-face

CSS declaration. Any TrueType/Open Type/Open Font Format file can be losslessly converted to WOFF for Web use (subject to licensing of the font data); once decoded by a user agent, the WOFF font will display identically to the original desktop font from which it was created.

The WOFF format also allows additional metadata to be attached to the file; this can be used by font designers to include licensing or other information, beyond that present in the original font. Such metadata does not affect the rendering of the font in any way, but may be displayed to the user on request.

The WOFF format is not expected to replace other formats such as TrueType/Open Type/Open Font Format or SVG fonts, but provides an alternative solution for use cases where these formats may be less performant, or where licensing considerations make their use less acceptable.

## Why Use WOFF

WOFF provides a lot of advantages over other font choices. The standard fonts you can use end up making a web page. WOFF fonts have the following benefits:

- They are more accessible than fonts as images. WOFF styles plain HTML text with CSS.
- WOFF files include typographical information like contextual forms and old style figures. This gives your web pages better typography because you're using the correct characters, not just approximations.
- WOFF fonts can help with internationalization because you can embed fonts with characters from other languages.
- Like all CSS styled text, fonts styled with WOFF are more search engine friendly.
- WOFF fonts are compressed, so they are smaller than other types of font files

**For more details:** <http://www.w3.org/TR/WOFF/>

### 4.3.3 Inscript Keyboard

What is required is for the input device to permit the entry of the complete range of input characters required in a particular language.

For India, the national standard is Inscript. It allows all characters of an Indian Language to be entered in accordance with Unicode. An enhanced version of Inscript is likely to be released soon where the same principles will apply.

#### Principles of Inscript keyboard:

- The division of the keyboard into Vowel and Consonant groups on the middle two rows (left half).
- The accommodation of vowels as independent and dependent (or as modifiers/markers/matras) and on the same key, where such distinction occurs.
- The accommodation of consonants on the basis of the “Vargas”/classes with unaspirates and aspirate on the same key.
- Across different languages, there is maximum similarity of characters being located on the same keys. Different positions were used only for characters specific to individual scripts/languages.

#### Guiding principles for Inscript keyboard layout

**Principle 1: Usability:** The backward compatibility with existing keyboard layout will be maintained as far as possible.

**Principle 2: Compatibility across major operating systems:** The Keyboard is designed such that it will be compatible across the board for all Operating Systems, ensuring that no new learning habits need be mastered when one emigrates from one system to another.

**Principle 3: Phonetic parallelism:** As far as possible the principle of phonetic typing has also been retained and a user familiar with one language say HINDI can easily type in GUJARATI since the key positions are the same (spelling and grammar notwithstanding).

**Principle 4: Bi-lingual facility:** The principle has been retained in the revised keyboard not only because of usability criteria but also because Indian languages often mix English and Indian language, not only for

word but especially for punctuation markers which are borrowed from Latin1

**Principle 5: Keyboards are for language:** The current version of Inscript is script-based. This made it difficult to address languages that are based on the same script like Marathi-Hindi-Konkani (Unicode Devanagari) and Assamese-Bengali (Unicode Bengali). This has been discarded and the new keyboards are exclusive to each language.

**Principle 6: Placement of new characters & visual pertinence:** Where individual characters that are added are applicable across the spectrum of all Indian Languages (e.g. Avagraha, Om); such characters are placed uniformly on the same position for all key-boards. Additional Consonants are accommodated on the consonant pad and additional vowels on the Vowel-pad.

**Principle 7: Extensibility:** The keyboard will be extensible to accommodate all new characters which may eventually be added by future versions of Unicode.

**Principle 8: Backward compatibility:** Although this is really not pertinent to keyboard design, since the keyboard impinges on storage, the design of the extended keyboard will be such that earlier data made with the older keyboards will be compatible with that produced by the new keyboard.

**Principle 9: Visual display:** The characters pertinent to a given script/language and present in the Unicode chart will be engraved on the keyboard to ensure ease of use.

#### 4.3.4 Phonetic Keyboard

Phonetic keyboards are ones that allow inputs in the Latin/English character set and transliterate the script into another based on their sounds or phonetic characteristic.. This can work between any two scripts.

There are several phonetic input methods for Indian Languages. But there is no standard method of input. i.e. there is no standard sequence of characters in one script that convert to the equivalent phonetic characters in another script.

There are two ways to use phonetic keyboards in Indian Language applications:

- Use what is provided on the client platform.
- Use APIs provided by vendors within user-interfaces.

Option b is common because Option a is not standard.

User interface designers can make their own choices until a standard form is available for Indian Languages.

#### **4.3.5 Statutory Requirement for New Rupee Symbol**

Govt. of India has recently approved the New Rupee Symbol ₹ and now finalizes the representation of rupee symbol in INSCRIPT Keyboard layout and the Qwerty Keyboard.

#### **QWERTY & INSCRIPT Keyboard layout:**

AltGR + 4 is finalized for the position of rupee symbol in keyboard.

#### **Merits of selecting AltGR + 4**

- It is placed in the same row where all other International Currency Symbol were present
- Good Mnemonic values with Dollar sign
- On US International keyboard, Currently Generic Currency symbol is present which will be replaced by Rupee Symbol ₹, which is being used by other country specific Keyboards for their currency symbol implementation.
- Enhance INSCRIPT Keyboards for Indian Languages required minimal change to accommodate new Rupee symbol at the above mentioned position.

## 5. Code Elements

### 5.1 Language identification and Negotiation BCP 47/RFC 5646

#### DEFINITION:

The language used when presenting information. In some contexts, it is possible to have information available in more than one language, or it might be possible to provide tools (such as dictionaries) to assist in the understanding of a language. Also, many types of information processing require knowledge of the language in which information is expressed in order for that process to be performed on the information; for example spell-checking, computer-synthesized speech, Braille, or high-quality print renderings.

#### GOALS:

Language Tags can be used by the Applications to deliver to users the most appropriate information. It is useful for accessibility, authoring tools, translation tools, font selection, page rendering, search, and scripting and is used by screen readers and accessibility as these applications are interested in output produced by them in different language mode.

#### CONCEPT OF MACRO LANGUAGES AND COLLECTIVE LANGUAGES:

Macro Languages: Some Languages are defined as macro languages covering either significantly different dialects or a net of very closely related languages. Two new terms were introduced in ISO 639-2 called Macro languages and Collective languages. The term Collective language is not continued in ISO 639-3. However, Macro language is still working.

Some of examples of Macro languages are: doi: is the ISO 639-3 language code for Dogri. There are 2 individual language codes assigned: dgo — Dogri, an individual language and xnr — Kangri, a dialect.

Collective Languages: A collective language code element is an identifier that represents a group of individual languages that are not deemed to be one language in any usage context. Collective languages and their ISO 639-2 codes are: bih - Bihari. This Language also has an ISO 639-1 code and him – Himachali. However, this is bad example because these two are not kept separate for the revision.

## FORMAT:

The Format of Language Tags is described in BCP 47 published by Internet Engineering Task Force (IETF). It describes the structure, content, construction, and semantics of language tags for use in cases where it is desirable to indicate the language used in an information object, how to register values for use in language tags and the creation of user-defined extensions for private interchange.

The Language Subtag Registry, maintained by IANA, lists the current valid public subtags. Language tags consist of only language subtag, or a language subtag and a region subtag. Subtags are not case sensitive, but the specification recommends using the same case as in the Language Subtag Registry, where region subtags are uppercase, script subtags are title case and all other subtags are lowercase. Syntax will be:

Language-Tag = langtag

**/ Private use ; private use tag**

**/ grandfathered ; grandfathered registrations**

**Langtag = (language ["-" script] ["-" region] \*("-" variant) \*("-" extension) ["-" private use])**

For Example, bn language code is used for Bengali language; Beng Script code is used for Bengali Script and IN is used as Region Code for India. So, to represent the whole Language tag we can declare it as Bn-Beng-IN (Bengali language uses Bengali Script in India.).

## 5.2 ISO 639:1, 2, 3, 5-Language Tag Standard

ISO 639 is the set of international standards that lists short codes for language names. It provides two set of three-letter alphabetic codes for the representation of names of languages. The ISO 639 code is divided into two parts i.e. Two-character code (639-1) Example: hi for Hindi, kn for Kannada and Three characters code (639-2, 639-3,639-5) Example: mar for Marathi, san for Sanskrit Used in ISO 639-1.

Table-1

PRESENT TABLE OF LANGUAGE TAGS IN ISO 6391, 2 AND 3

| S.NO | 639-1 | 639-2 | 639-3 | Language Name   | Native Name    | Language Status                             |
|------|-------|-------|-------|-----------------|----------------|---|
| 1.   | as    | Asm   | asm   | <b>Assamese</b> | অসমীয়া        | <b>Scheduled (Assam)</b>                    |
| 2.   | -     | Sit   | brx   | <b>Bodo</b>     |                | <b>Scheduled (Assam)</b>                    |
| 3.   | bn    | Ben   | ben   | <b>Bengali</b>  | বাংলা          | <b>Scheduled (West Bengal, Bangladesh)</b>  |
| 4.   | -     | -     | dgo   | <b>Dogri</b>    | ڳوٺ / گوٹ      | <b>Scheduled (J&amp;K)</b>                  |
| 5.   | gu    | Guj   | guj   | <b>Gujarati</b> | ગુજરાતી        | <b>Scheduled (Gujarat)</b>                  |
| 6.   | kn    | Kan   | kan   | <b>Kannada</b>  | ಕನ್ನಡ          | <b>Scheduled (Karnataka)</b>                |
| 7.   | ks    | Kas   | kas   | <b>Kashmiri</b> | كشميري         | <b>Scheduled (POK )</b>                     |
| 8.   | mr    | Mar   | mar   | <b>Marathi</b>  | मराठी          | <b>Scheduled Maharashtra)</b>               |
| 9.   | ne    | Nep   | nep   | <b>Nepali</b>   | नेपाली         | <b>Scheduled (Nepal, Sikkim and Bengal)</b> |
| 10.  | or    | Ori   | ori   | <b>Oriya</b>    | ଓଡ଼ିଆ          | <b>Scheduled (Orissa)</b>                   |
| 11.  | pa    | Pan   | pan   | <b>Panjabi</b>  | ਪੰਜਾਬੀ         | <b>Scheduled (Punjab)</b>                   |
| 12.  | sd    | Snd   | snd   | <b>Sindhi</b>   | سنڌي،<br>سندهي | <b>Scheduled (Maharashtra, Gujrat)</b>      |
| 13.  | ta    | Tam   | tam   | <b>Tamil</b>    | தமிழ்          | <b>Scheduled (Tamil Nadu)</b>               |
| 14.  | te    | Tel   | tel   | <b>Telugu</b>   | తెలుగు         | <b>Scheduled (A.P)</b>                      |
| 15.  | ur    | Urd   | urd   | <b>Urdu</b>     | اردو           | <b>Scheduled (A.P, Delhi, J&amp;K)</b>      |
| 16.  | sa    | San   | san   | <b>Sanskrit</b> | सं कृतम्       | <b>Scheduled</b>                            |
| 17.  | -     | Kok   | Gom   | <b>Konkani</b>  | कोंकणी         | <b>Scheduled (Goa)</b>                      |

|     |    |     |       |                        |               |                              |
|-----|----|-----|-------|------------------------|---------------|------------------------------|
|     |    |     | (kok) |                        |               |                              |
| 18. | -  | Mai | mai   | <b>Maithili</b>        | मैथिली        | <b>Scheduled (Bihar)</b>     |
| 19. | -  | Mni | mni   | <b>Manipuri/Meetei</b> | মেইতেই<br>লোন | <b>Scheduled (Manipur)</b>   |
| 20. | -  | Sat | sat   | <b>Santhali</b>        |               | <b>Scheduled (Jharkhand)</b> |
| 21. | hi | Hin | hin   | <b>Hindi</b>           | हिन्दी        | <b>Scheduled (UP, UK)</b>    |
| 22. | ml | Mal | mal   | <b>Malayalam</b>       | മലയാളം        | <b>Scheduled</b>             |

**For more details:** <http://www.w3.org/International/articles/language-tags/>

**For more details:** <http://www.sil.org/iso639-3/default.asp>

### 5.3 CSS

Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation semantics (the look and formatting) of a document written in a mark-up language. Its most common application is to style web pages written in HTML and XHTML, but the language can also be applied to any kind of XML document, including SVG and XUL.

CSS is designed primarily to enable the separation of document content (written in HTML or a similar mark-up language) from document presentation, including elements such as the layout, colors, and fonts.[citation needed] This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for tableless web design). CSS can also allow the same mark-up page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille- based, tactile devices. While the author of a document typically links that document to a CSS style sheet, readers can use a different style sheet, perhaps one on their own computer, to override the one the author has specified.

## Syntax:

- CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties.
- A style sheet consists of a list of rules. Each rule or rule-set consists of one or more selectors and a declaration block. A declaration-block consists of a list of declarations in braces. Each declaration itself consists of a property, a colon (:), a value, then a semi-colon (;).
- In CSS, selectors are used to declare which of the mark-up elements a style applies to, a kind of match expression. Selectors may apply to all elements of a specific type, or only those elements that match a certain attribute; elements may be matched depending on how they are placed relative to each other in the mark-up code, or on how they are nested within the document object model.
- Pseudo-classes are another form of specification used in CSS to identify mark-up elements, and in some cases, specific user actions to which a particular declaration block applies. An often-used example is the :hover pseudo-class that applies a style only when the user 'points to' the visible element, usually by holding the mouse cursor over it. It is appended to a selector as in a :hover or #element id: hover. Other pseudo-classes and pseudo-elements are, for example, :first-line, :visited or :before
- A pseudo-class selects entire elements, such as :link or :visited, whereas a pseudo-element makes a selection that may consist of partial elements, such as :first-line or :first-letter.
- Selectors may be combined in other ways too, especially in CSS 2.1, to achieve greater specificity and flexibility

Here is an example summing up the rules above:

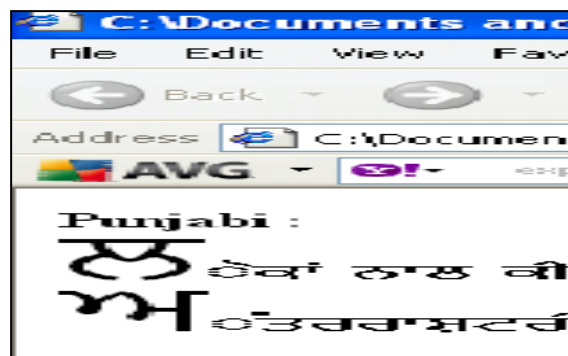
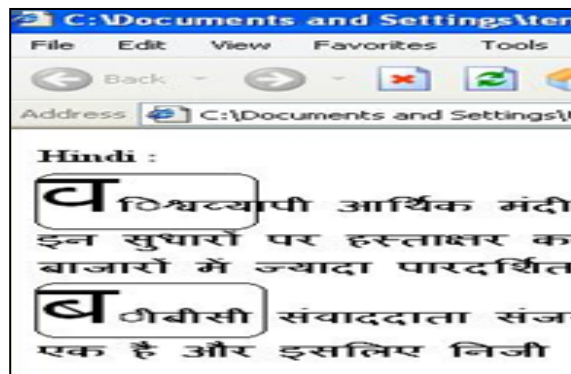
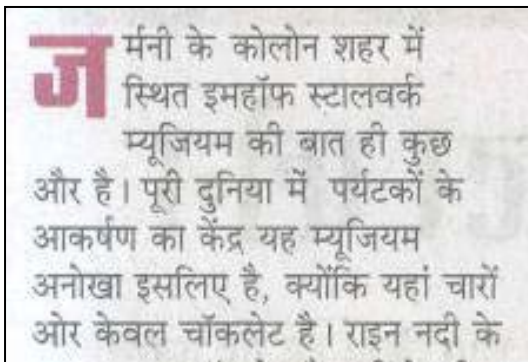
```
Selector [, selector2 ...][:pseudo-class] {
Property: value;
[property2: value2;
...]
}
/* comment */
```

## Styling Issues in Indian Languages:

### 1. Styling of first letter pseudo-element Issues in Indian script – Hindi

The first-letter pseudo-element represents the first letter of the first line of a block, if it is not preceded by any other content (such as images or inline tables) on its line. It allows that first letter to be styled individually, without markup. It may be used for "initial caps" and "drop caps"

For more details: [www.w3.org/TR/CCS2/Selector.html](http://www.w3.org/TR/CCS2/Selector.html)



This example is showing a possible rendering of an initial cap. In this example the computed height or width of the first letter is different, that is why it is not showing first character properly. (IE, Opera)

## 2. Bullets and Numbers

- Number schemes/ bulleting needs to be supported in Indian languages as well. Some standards however need to be provided to those developing CSS so that by default user could have the facility to use bulleting in his own Indic languages.
- The word processors are sometimes used by the user to develop pages for the web. Therefore standards are must. In most application Devanagari order is followed for languages sharing the script, which unfortunately is not the correct thing while deciding on sorting/collation for Indic languages. Number schemes to be supported in Indian languages also.
- In case of Bangla and Assamese language the bulleting should be like as in below In case of numeric bulleting the digit of the Bangla scripts like ১,২,৩,৪,৫,৬,৭,৮,৯,০ is to be used.
- For bulleting by alpha numeric the consonant of Bangla scripts are used. The Bangla vowels are not used in bulleting purpose.

|     |        |
|-----|--------|
| अ ) | U+0905 |
| आ ) | U+0906 |
| इ ) | U+0907 |
| ई ) | U+0908 |
| उ ) | U+0909 |
| ऊ ) | U+090A |
| ए ) | U+090F |
| ऐ ) | U+0910 |
| ओ ) | U+0913 |
| औ ) | U+0914 |
| ... |        |

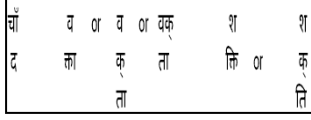
Hindi

|     |        |
|-----|--------|
| ੳ ) | U+0A73 |
| ਅ ) | U+0A05 |
| ੲ ) | U+0A72 |
| ਸ ) | U+0A38 |
| ਰ ) | U+0A39 |
| ਕ ) | U+0A15 |
| ਖ ) | U+0A16 |
| ਗ ) | U+0A17 |
| ਘ ) | U+0A18 |
| ਙ ) | U+0A19 |
| ਚ ) | U+0A1A |
| ਛ ) | U+0A1B |
| ... |        |

Punjabi

### 3. Vertical & Horizontal Alignment arrangements of characters

Presentation / Styling issues: Vertical arrangement of characters If some string is written in vertical mode, then writing each character on a new line may not be suitable.



In case of horizontal alignment of characters, the space is given between the every character in case of English. But in case of Indian language like Bangla, Assamese etc the space may given not in every character but after some portion of the character sequence as in above figure.

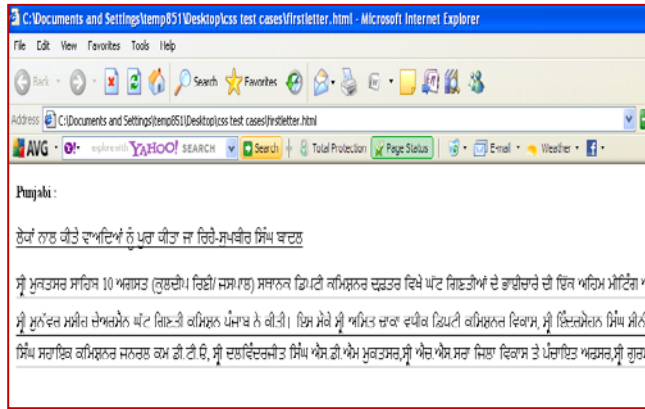
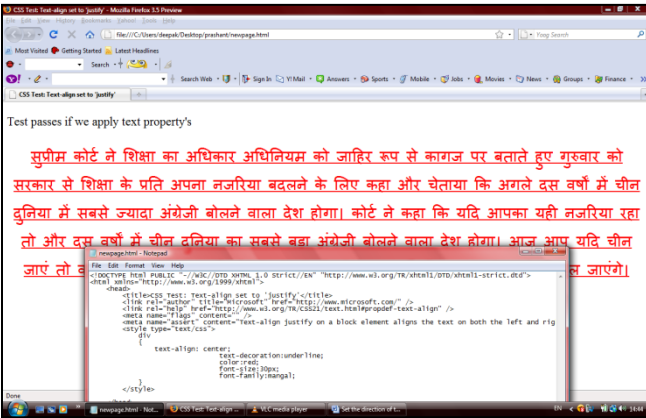
Horizontal justification in HTML page in Indic languages, does not works on few browsers. It works well on IE6.0 but not on Firefox 2.

### 4. Underlining of the characters

- There is some examples of Indian languages in which Matra's are not readable due to

#### Underlining of characters:

- Hindi - अन्य भाषाओं में भी अनुवाद
- Punjabi ਗੁਰੂ
- Bengali :তাই পুরোনো আর্কাইভ একটু ওলট পালট।
- Gujarati - સરદાર ગુજરી
- -Marathi- मराठी मुला मुलींची नावे



For more details: [www.w3.org/TR/CCS2/text.html](http://www.w3.org/TR/CCS2/text.html)

### 3. Over lining of the characters

Whenever we use special characters in Hindi and applying text decoration as over line we are finding some issues in almost every India language where we are using any special characters. Issues w.r.t. languages like Hindi, bangle, Punjabi, Malayalam, Tamil, Oriya

### 5.4 XML (Extensible Mark-up Language)

XML is a big open-standards victory for users.

- **Freely extensible**
  - No tag name limitations
  - No language limitations
- **Human-readable**
  - Can maintain data using basic text tools like sed, awk, perl, Word macros
- **Open standard**
  - In theory, XML users can't be held hostage to vendor control

- **Easy to implement**

- There will be many powerful, cheap, off-the-shelf commercial XML tools
- There is already an ever-growing set of free XML tools (almost all of them Java-based)

## **Requirement of XML for Indian Language:**

### **Internationalization Tag Set (ITS)**

- ITS is a technology to easily create XML which is internationalized and can be localized effectively.

### **ITS for Schema developers**

- User will find proposals for attribute and element names to be included in their new schema (also called "host vocabulary").
- It leads to easier recognition of the concepts represented by both schema users and processors.

### **Main Attributes**

- Defining mark-up for natural language labelling.
- Defining mark-up to specify text direction.
- Indicating which elements and attributes should be translated.
- Providing information related to text segmentation.
- Defining mark-up for unique identifiers.
- Defining mark-up for notes to localizers.
- Working with multilingual documents.

## Advantage of XML over Database

- **Built in support** - for internationalization due to the fact that it utilizes Unicode.
- **Platform independence**- for instance, no needs to worry about endianness.
- **Readable** - Human readable format makes it easier for developers to locate and fix errors than with previous data storage formats.
- **Extensibility** - XML allows developers to add extra information to a format without breaking applications.
- **Support off-the-shelf tools** - Large number of off-the-shelf tools for processing XML documents already exists.

## 5.5 XHTML

**XHTML** (Extensible Hypertext Mark-up Language) is a family of XML mark-up languages that mirror or extend versions of the widely used Hypertext Mark-up Language (HTML), the language in which web pages are written.

While HTML (prior to HTML5) was defined as an application of Standard Generalized Mark-up Language (SGML), a very flexible mark-up language framework, XHTML is an application of XML, a more restrictive subset of SGML. Because XHTML documents need to be well-formed, they can be parsed using standard XML parsers—unlike HTML, which requires a lenient HTML-specific parser.

XHTML 1.0 became a World Wide Web Consortium (W3C) Recommendation on January 26, 2000. XHTML 1.1 became a W3C Recommendation on May 31, 2001. XHTML5 is undergoing development as of September 2009, as part of the HTML5 specification.

## Versions of XHTML:-

### XHTML 1.0

This was the first XHTML standard to be released. It was created to redesign HTML 4.0 to act more like an XML file. To make the change from HTML 4.0 to XHTML 1.0, there were three standards: **strict**, **transitional** and **frameset**.

- Strict was the standard that accepted the full XHTML standard.
- Transitional was used by developers to either make it easier to migrate over from HTML 4.0 or to allow certain standards from HTML version 4.01 Transitional in the XHTML file.
- Frameset was very simply the standard that allowed frames in the XHTML file.

XHTML has the capability to be used with XML files and applications that work with XML files.

### XHTML 1.1

XHTML 1.0 was made with the intention of allowing developers to move from HTML standards to XHTML standards more easily.

### XHTML 2.0

XHTML 2.0 was drafted between August 2002 and July 2006. It was designed to supersede HTML 4.0 and XHTML 1.1. In July 2009, World Wide Web Consortium announced plans to discontinue work on XHTML 2.0 by the end of the year in order to focus its effort on HTML 5. HTML 5 is expected to include a XML scheme in place of XHTML 2.0 currently called XHTML 5.

## Converting a document from HTML 4.0 to XHTML 1.0:-

Converting a document from HTML 4.0 to XHTML 1.0 will not be a totally painless affair - some changes WILL need to be made.

- **An XHTML document MUST be well-formed XML**  
It must conform to basic XML syntax. If it does not, the XML parser does not have an obligation to continue processing the document. Unlike today's HTML parsers, an XML parser will not try to recover and "guess" what you meant if the syntax is incorrect.

- **<Html> MUST be the top-level element.**  
Not changes from HTML, but there are quite a few documents out there that neglect this important point.
- **Element and attribute names MUST be in lower case**  
HTML is not case-sensitive; but XML is.
- **Attribute values MUST be quoted**
- **End tags are required for non-empty elements.**  
They are no longer optional.  
Affected Elements: base font, body, colgroup, dd, dt, head, html, li, p, rt, spacer, tbody/thead/tfoot, th/td, tr
- **All empty elements must use the XML "empty tag" syntax**  
XML empty elements are explicitly closed with a trailing forward slash ("/") before the end bracket (eg: <br> becomes <br />)  
Affected Elements: area, base, bgsound, br, col, frame, hr, img, input, isindex, keygen, link, meta, option, param, wbr
- **XML does not allow attribute minimization.**  
Stand-alone attributes must be expanded (eg: <td nowrap>cell</td> becomes <td nowrap="nowrap">cell</td>)
- **Whitespace handling in attribute values is different in XML.**  
Leading/trailing spaces are truncated, and multiple spacing characters within the attribute value are collapsed to single spaces.
- **Script sections should be wrapped in XML CDATA sections**
- **SGML DTD exclusions are not possible in XML, but they should still be observed as "good practice".**  
Not allowed to nest within themselves: a, button, form, label  
Pre exclusions: big, img, object, small, sub, sup  
Button exclusions: fieldset, form, iframe, input, label, select, text area

## XHTML Document Type Definition (DTDs):-

There are three kinds of standard XHTML DTDs:

1. Strict
2. Transitional (also called *loose*)
3. Frameset

Each of these XHTML DTDs targets a different level of detail for XHTML. In other words, one can choose the kind of feature one want from the different features supported by these standard XHTML DTDs.

If the page, for instance, does not require an advanced feature of XHTML, we can use a minimal DTD. Keep in mind smaller DTDs result in faster validation, which in turn, results in a faster display of the web page.

A description for each of these standard XHTML DTDs is given in the table below.

- The strict DTD is the most efficient as it provides minimal XHTML language for creating web pages. Actually it enables a fastest validation of XHTML documents. The idea behind using a strict DTD is to use style sheets for display rather than presentation elements. Thus use of a strict DTD aims to separate presentation code from content.
- The transitional adds more features than a strict DTD to a XHTML document. The support for more features increases the validation process when an XHTML document is displayed. Because a transitional DTD provides support for presentation elements, the transitional DTD can be used for converting HTML documents to XHTML. The transitional DTD does not include support for frames (which allows us to use a single browser window to display multiple web pages)
- The third DTD, frameset, is a full-featured XHTML DTD. This DTD has support for what a transitional DTD offers plus support for frames. Because frameset DTD is the most complex, it is the slowest of three when it comes to validating XHTML documents.

| Table : Summary of the three XHTML DTDs  |
|--|
| Description  |
| The <i>strict</i> type DTD does not support any HTML presentation elements (such as <p>, <a>, <b>, etc.). This is the <i>low-featured</i> XHTML DTD.   |
| The <i>Transitional</i> type DTD adds support for HTML presentation elements. That means you can use HTML elements (such as <p>, <a>, <b>, etc.) directly inside of your XHTML document. The Transitional DTD is a <i>medium-featured</i> XHTML DTD. |
| The <i>Frameset</i> type DTD adds support for frames. This is the most <i>full-featured</i> XHTML DTD.   |

For more details : <http://www.w3.org/TR/xhtml1/>

## Issues in Indian languages:

### Browser Dependency – Display problem in some browser

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="hi" lang="hi">

<head>
<meta http-equiv="Content-Type" content="text/xhtml+xml; charset=utf-8"/>
<meta http-equiv="Content-Language" content="hi" />
<title>हिन्दी </title>
</head>
<body bgcolor="#FFFFFF" link="#000000" text="red">
<p>वाराणसी, जागरण टीम। जिले में जहरीली शराब पीने से बुधवार को 16 लोगों की मौत हो गई। एक मौत मंगलवार रात हो गई थी। दो दर्जन लोग विभिन्न अस्पतालों में इलाज करा रहे हैं। पुलिस ने मामले की जांच शुरू कर दी है। इसमें अभिसूचना इकाई की टीम को भी लगाया गया है। सर्वाधिक मौतें सोएपुर (किंठ) में हुई हैं। यहां मृत लोगों ने गांव में ही रहने वाली एक महिला से शराब खरीदी थी। अकथा [सारनाथ] में दो मौतें हुईं। इसी थाना क्षेत्र के बेनीपुर इलाके में दो अन्य लोगों की जान गई।</p>
</body>
</html>

```



XHTML Code

वाराणसी, जागरण टीम। जिले में जहरीली शराब पीने से बुधवार को 16 लोगों की मौत हो गई। एक मौत अस्पतालों में इलाज करा रहे हैं। पुलिस ने मामले की जांच शुरू कर दी है। इसमें अभिसूचना इकाई की र [किंठ] में हुई हैं। यहां मृत लोगों ने गांव में ही रहने वाली एक महिला से शराब खरीदी थी। अकथा [सारन इलाके में दो अन्य लोगों की जान गई।

Less vertical alignment between last two lines



Tested in IE

वाराणसी, जागरण टीम। जिले में जहरीली शराब पीने से बुधवार को 16 लोगों की मौत हो गई। एक मौत मंगलवार रात हो गई थी। दो दर्जन लोग विभिन्न अस्पतालों में इलाज करा रहे हैं। पुलिस ने मामले की जांच शुरू कर दी है। इसमें अभिसूचना इकाई की टीम को भी लगाया गया है। सर्वाधिक मौतें सोएपुर (किंठ) में हुई हैं। यहां मृत लोगों ने गांव में ही रहने वाली एक महिला से शराब खरीदी थी। अकथा [सारनाथ] में दो मौतें हुईं। इसी थाना क्षेत्र के बेनीपुर इलाके में दो अन्य लोगों की जान गई।

Alignment is Proper



Tested in Mozilla

## 5.6 X-Forms

### What are X-Forms?

- A revision to the existing HTML forms technology developed by the World Wide Web Consortium.
- X-forms is XML based Open Standard for e-forms domain
  - XML is the de-facto standard for data and applications interoperability
- Separates Data, Logic and Presentation
- Uses Declarative language by avoiding scripting
- Design started from scratch Based on totally new model
- Not compatible with HTML Forms

1993 : HTML forms

1994–2001: Few non-proprietary extensions

1997 : XML Schema and standard data types

2000 : CSS

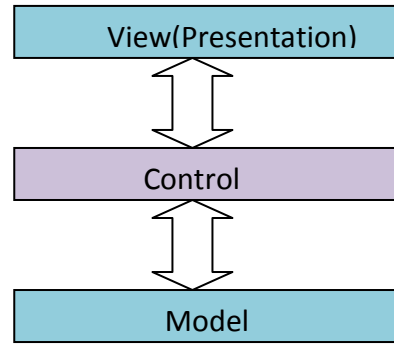
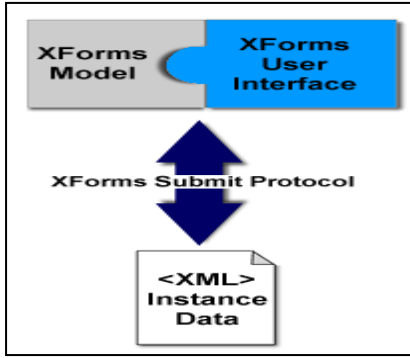
2002 : X-Forms initial drafts

2006 : World Wide Web Consortium (W3C) X-Forms 1.0  
“Recommendation” status

2009 : World Wide Web Consortium (W3C) X-Forms 1.1  
“Recommendation” status

### X-forms: MVC

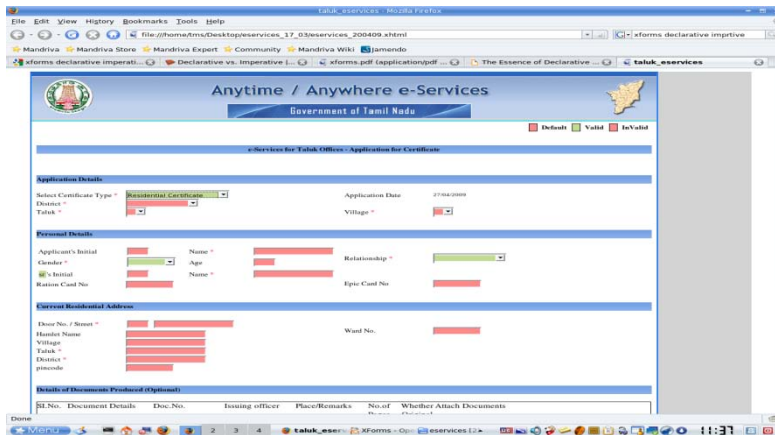
- “Model” describes form data, constraints and submission.
- “View” describes what visual controls appear in the form, how they are grouped together and what data they are bound to. CSS can be used to describe a form's appearance
- “Controller” that handles event input and the mappings between the two



## X-forms Framework:

- Based on MVC paradigm
- X-Forms model defines
  - What the form is
  - What data it contains
  - What it should do
- X-Forms user interface defines the form controls and how they should be displayed
- Instance contains the data and initial form values
- Submit protocol defines how X-Forms sends and receives instance data
- Bindings bind properties to instance data

## X-Forms



## X-forms Implementations:

1. Native browser support
  - Browser is able to interpret and display XHTML+X-Forms documents as such E.g., X-Smiles
2. Browser plug-in
  - User has to install the plug-in once
  - E.g., Mozilla X-Forms for Firefox, MozzIE for IE

3. Ajax implementation
    - A JavaScript component, which is embedded into XHTML+X-Forms document
    - E.g., forms Player for IE, Google's Ubiquity, form faces for major browsers
  4. Server-side transformation
    - XHTML+X-Forms is transformed to HTML+ CSS+ Java script
- E.g.
- . Orbeon (CMS like Life Ray uses Orbeon)
  - . Chiba (CMS like Alfresco uses Chiba)

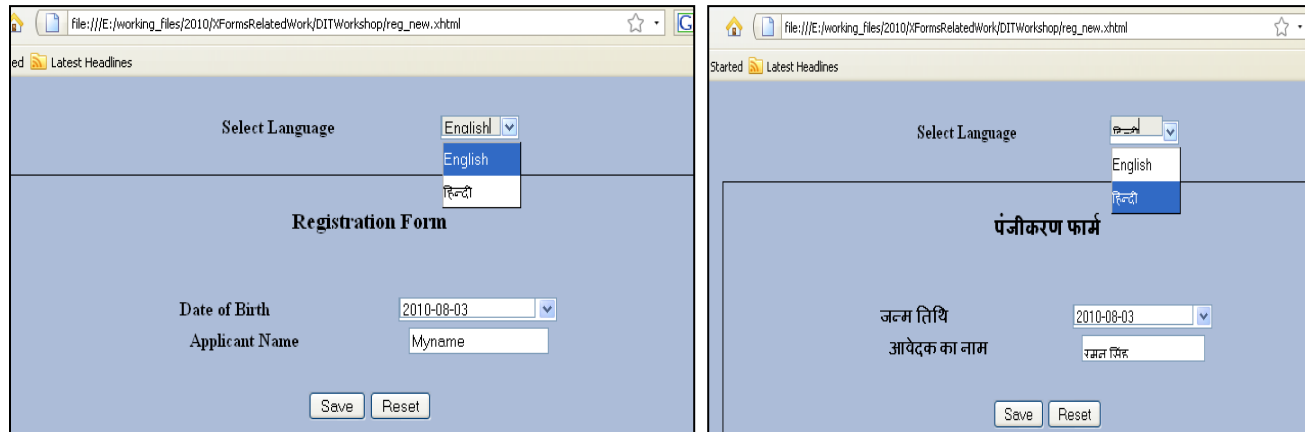
### **Unicode with X-forms:**

- 1) X-Forms use XML as its underlying serialization form.
- 2) Any browser uses XML while receiving or sending X-forms from server to client and vice-versa
- 3) XML documents are Unicode.
- 4) They can be stored in other encodings, but the translation between those encodings and Unicode is simple and deterministic.
- 5) This makes it possible to handle data in various international as well as national languages.

### **Internationalisation and Localisation in X-forms**

- 1) True internationalization and localization go beyond merely letting people type their names and addresses using non-ASCII letters.
- 2) It's also necessary to allow forms in languages such as Hebrew and Arabic to flow right to left rather than left to right.
- 3) To handle this again, the separation of presentation from content played an important role.
- 4) Since presentation and contents are totally separate in X-Forms so fields can be laid in the direction, based on the requirement of local renderer.

## Examples displaying Unicode support in X-Forms



## Need of X-forms in E-governance applications

The needs of X-form in E-Governance application are:

### 1) Increased Interoperability

- Standardized forms interoperability (with business partners, customers, suppliers) enables process/value chain integration
- ISV application processing interoperability between vendors provides customers with flexibility and choice supporting heterogeneous environments.

### 2) Faster Time-to-Market and Reduced Development Costs

- X-Forms embeds forms data processing rules using an open standard
- Standard X-Form Processors can be used instead of creating custom code for each form
- Problem Today: Every bank processing a mortgage form must create custom code to ensure that the hundreds of application rules are processed correctly – ex. Spousal info

- Solution: An X-Forms mortgage application embeds the rules that describe when spousal information is relevant for application. Ex. Joint mortgage applications only

### 3) Reusable Form Components leveraged by Industry

- Forms Standards enables libraries of reusable standard forms and form components

### 4) Multi-platform Support

- UI standardization provides a model to support multi-platform, accessibility, localization and roles within forms

### 5) Enhances and Complements SOA

- X-Forms provides a Forms Data Processing Model & supports active content using declarative rules & Web Services

For more details: <http://www.w3.org/TR/xforms/>

## 5.7 HTML 5.0

HTML4 became a W3C Recommendation in 1997. While it continues to serve as a rough guide to many of the core features of HTML, it does not provide enough information to build implementations that interoperate with each other and, more importantly, with a critical mass of deployed content. The same goes for XHTML1, which defines an XML serialization for HTML4, and DOM Level 2 HTML, which defines JavaScript APIs for both HTML and XHTML. HTML5 will replace these documents.

HTML5 defines an HTML syntax that is compatible with HTML4 and XHTML1 documents published on the Web, but is not compatible with the more esoteric SGML features of HTML4. W3C HTML5 defines:

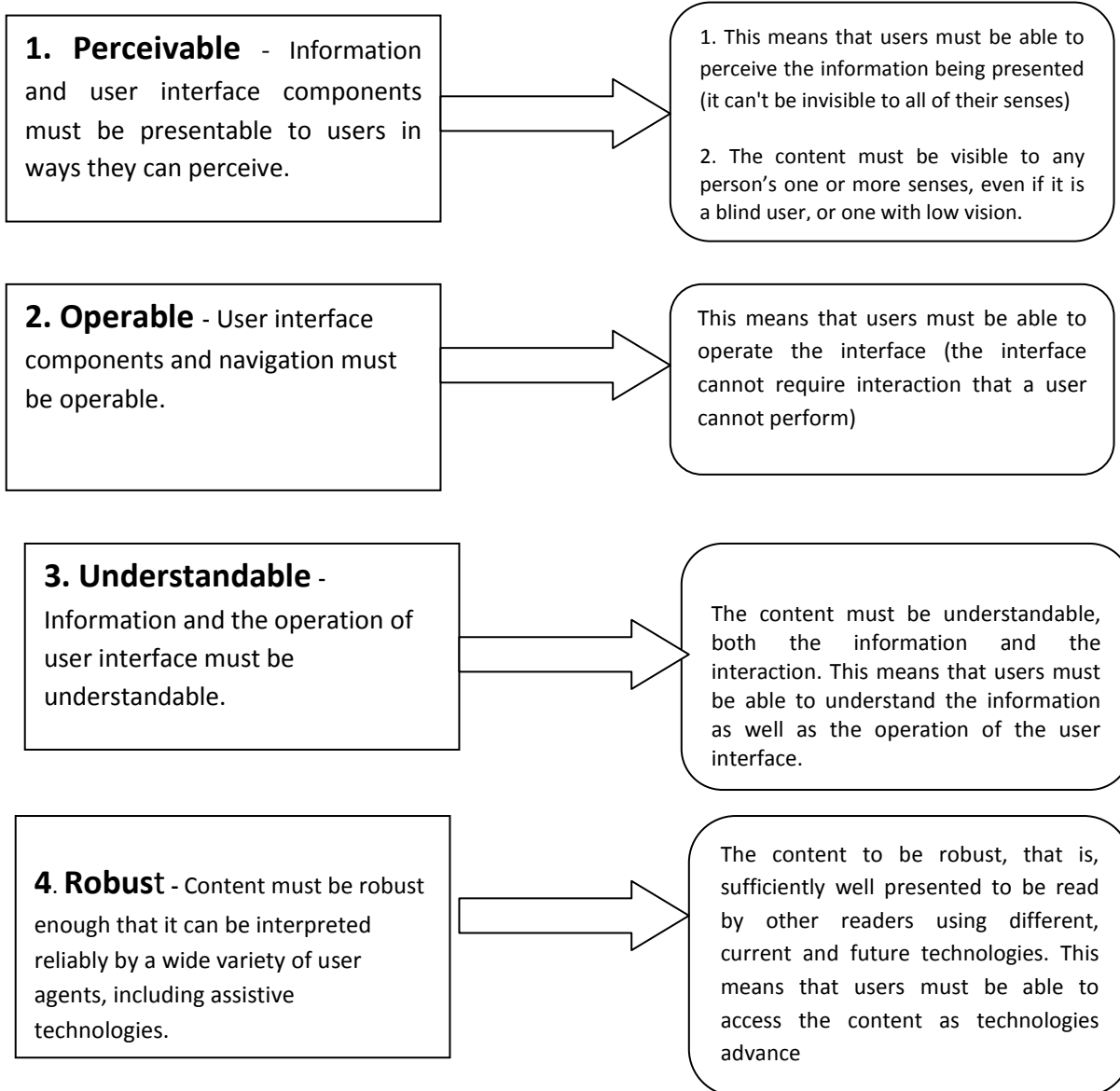
1. A single language called HTML5 which can be written in HTML syntax and in XML syntax.
2. A detailed processing model to foster interoperable implementations.
3. How to Improves mark-up for documents.
4. APIs for emerging idioms, such as Web applications.
5. Detailed parsing rules (including "error handling")

For more details : <http://www.w3.org/TR/2011/WD-html5-diff-20110525/>

## 5.8 Web Accessibility

Web Content Accessibility defines how to make Web content more accessible to people with disabilities. Accessibility involves a wide range of disabilities, including visual, auditory, physical, speech, cognitive, language, learning, and neurological disabilities. Although these guidelines cover a wide range of issues, they are not able to address the needs of people with all types, degrees, and combinations of disability

The top are four principles that provide the foundation for Web accessibility: perceivable, operable, understandable, and robust.



If any of these are not true, users with disabilities will not be able to use the Web.

## Layers of Guidance

The 12 guidelines are basic goals that authors of Web content should work toward in order to create accessible content. None of them are testable and are only meant as a framework of overall objectives.

- i. **Text Alternatives:** To include alternative text for images. When you make the decision to add alternative text, you include the many people who use talking browsers, screen readers, text browsers or browsers on small devices.
  - **Non-text Content:** All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations listed below.
  - **Controls Input:** All non-text content that is presented to the user has a text alternative that serves the equivalent purpose, except for the situations.
  - **Time-Based Media:** If non-text content is time-based media, then text alternatives at least provide descriptive identification of the non-text content.
  - **Test:** If non-text content is a test or exercise that would be invalid if presented in text, then text alternatives at least provide descriptive identification of the non-text content.
  - **Sensory:** If non-text content is primarily intended to create a specific sensory experience, then text alternatives at least provide descriptive identification of the non-text content.
  - **CAPTCHA :** If the purpose of non-text content is to confirm that content is being accessed by a person rather than a computer, then text alternatives that identify and describe the purpose of the non-text content are provided, and alternative forms of CAPTCHA using output modes for different types of sensory perception are provided to accommodate different disabilities.
  - **Decoration, Formatting, Invisible:** If non-text content is pure decoration, is used only for visual formatting, or is not presented to users, then it is implemented in a way that it can be ignored by assistive technology

ii. **Time-based Media: Provide alternatives for time-based Media.**

- **Audio-only and Video-only (Pre-recorded)**

An alternative for time-based media or audio description of the pre-recorded video content is provided for synchronized media, except when the media is a media alternative for text and is clearly labelled as such.

**Pre-recorded Audio-only:** An alternative for time-based media is provided that presents equivalent information for pre-recorded audio-only content.

**Pre-recorded Video-only:** Either an alternative for time-based media or an audio track is provided that presents equivalent information for pre-recorded video-only content.

- **Captions (Pre-recorded):**

Captions are provided for all pre-recorded audio content in synchronized media, except when the media is a media alternative for text and is clearly labelled as such. (Level A)

There are two exceptions when captions are not required:

- When the audio does not add any further information to the text displayed on the web page.
- When the audio is already an alternative for another presentation.

iii. **Audio Description or Media Alternative (Pre-recorded):**

An alternative for time-based media or audio description of the pre-recorded video content is provided for synchronized media, except when the media is a media alternative for text and is clearly labelled as such.

- **Captions (Live) (Level AA)**

Captions are provided for all live audio content in synchronized media.

- **Audio Description (Pre-recorded)**

Audio description is provided for all pre-recorded video content in synchronized media.

- **Sign Language (Pre-recorded):**

Sign language interpretation is provided for all pre-recorded audio content in synchronized media.

- **Extended Audio Description (Pre-recorded):**

Where pauses in foreground audio are insufficient to allow audio descriptions to convey the sense of the video, extended audio description is provided for all pre-recorded video content in synchronized media.

**iv. Adaptable:**

Create content that can be presented in different ways (for example simpler layout) without losing information or structure.

**v. Distinguishable:**

Make it easier for users to see and hear content including separating foreground from background.

**vi. Keyboard Accessibility:**

Make all functionality available from a keyboard.

**vii. Enough Time :**

Provide users enough time to read and use content

**viii. Seizures:**

Do not design content in a way that is known to cause seizures.

**ix. Navigable:**

Provide ways to help users navigate, find content, and determine where they are.

**x. Readable:**

Make text content readable and understandable.

**xi. Predictable:**

Make Web pages appear and operate in predictable ways.

**xii. Input Assistance :**

Help users avoid and correct mistakes.

**xiii. Compatible :**

Maximize compatibility with current and future user agents, including assistive technologies.

## **5.9 E-governance as Web Services**

The goal of the Web Services Internationalization Task Force is to ensure that Web Services have robust support for global use, including all of the world's languages and cultures.

The goal of this document is to examine the different ways that language, culture, and related issues interact with Web Services architecture and technology. Ultimately this will allow us to develop standards and best practices for those interested in implementing internationalized Web Services. We may also discover latent international considerations in the various Web Services standards and propose solutions to the responsible groups working in these areas.

Web services have emerged as the next generation of Web-based technology for interoperability. Web services are modular, self-describing, self-contained applications that are accessible over the Internet. Based on open standards, Web services enable constructing Web-based applications using any platform, object model, and programming language. A service is a collection of operations accessible through an application programming interface that allows users to invoke a service, which could be a response to a simple request to create a map

or a complicated set of image-processing operations running on several computers.

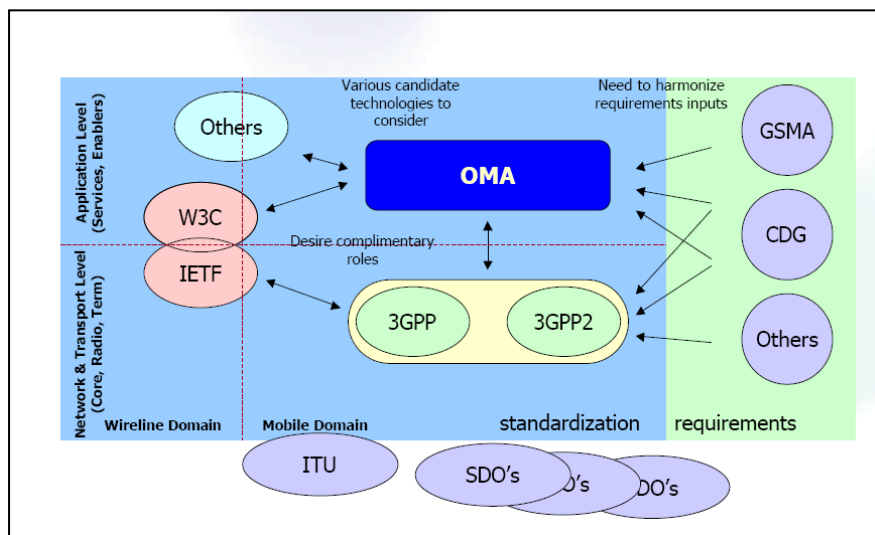
#### **Four important facts about Web services:**

- Web services are based on standard protocols (XML, SOAP, WSDL) developed by industry leaders, including Microsoft, and submitted to independent public standards organizations like the World Wide Web Consortium (W3C). Use of these standards enables interoperability.
- Web services interoperate by sending discrete messages; they do not need the continuous network connection that makes traditional systems vulnerable to network downtime.
- Web services share data and functionality; unlike the static server-client model that has dominated the Web, where Web pages present "snapshots" of data, the Web service model is dynamic. Large tasks can be distributed over multiple computers interacting and operating in tandem.
- An application that is "exposed" (made available for use) as a Web service can be "consumed" (used) by any application that can read XML, no matter what platform or device the application is running on. The same Web service can be used by a PC, a laptop, or a PDA—any "smart client"—so your developers don't need to program new and separate applications for each device.

**For more details: <http://www.w3.org/WAI/intro/accessibility.php>**

## 5.10 Mobile Web and its implementation

India continues to be one of the world’s fastest-growing telecommunications market due to a progressive regulatory regime, huge capital outlays for network expansion by operators, reductions in tariffs and cost of handsets. For the India mobile market, voice services constitute nearly 88 percent of the total revenue. For non-voice or data services which include SMS, Internet browsing and multimedia, year-on –year growth was 33 percent over 2008. Mobile media has increasingly becoming the preferred medium of Communication. Indian Mobile Market is one of the fastest growing – has overtaken China in terms of growth. Usage of Mobile Web is growing besides the messaging services.

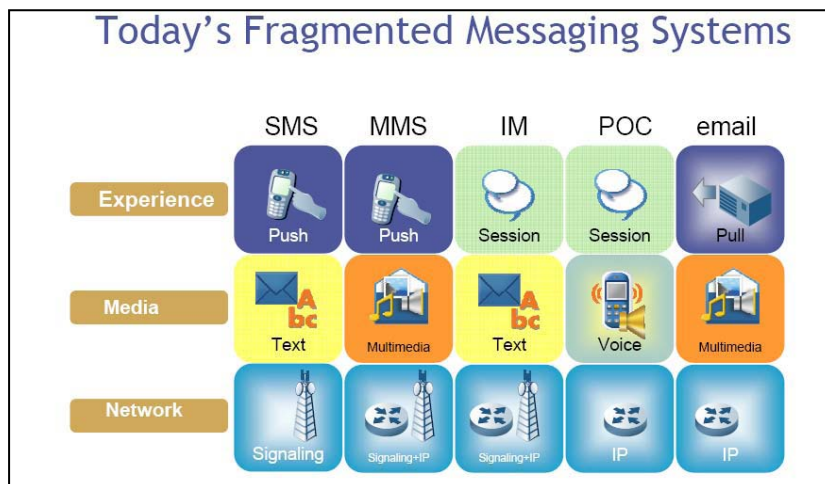


### Fragmentation of Mobile Standards and their Fall-outs:

- Lack of Common Industry view on architectural framework
- Lack of uniform interoperability plans and applications
- Standardization work driven by different technology frame works and not by services and applications.

## Fallouts:

- Lack of multi standard interoperability detracts from the mobile consumer experience.
- Impacts innovation in services and applications
- Increases costs for all involved.
- Slows down third-party content and application.



## Limitations of Mobile Devices

### Restrictions:

- Small memory, slower processors, small bandwidth
- These are cited often, but less and less relevant
- Modern mobiles are like PC-s 10 years ago and getting stronger every day

### Constraints:

- Small overall size, screen and keyboard (or no keyboard at all)
- Different usage environments (on the street, on a train...)
- Hands-free, one handed, or two handed usage
- Support of Unicode is increasing on mobile phones But, the problems come with displaying complex Languages, like Indic Languages

- Most devices are first designed for simple left to right text
- Later the software is converted to handle multilingual text

## **Complexity and Challenges for Indic Languages**

- Large linguistic diversity with 22 officially recognized languages and 12 scripts.
- One-language Many Scripts; Many Languages–One Script
- Specificity for each language and script is unique in nature and cannot be easily replicated, even if they share common characteristics
- Difference in perceptions of usage among various stakeholders, e.g. State Governments, Academia and industry
- Some of the languages have coverage across different nations across SAARC countries.
- Involves inter disciplinary research in advanced and sophisticated computer processing involving Artificial Intelligence and Machine Learning one hand; linguistic knowledge for incorporating human communication techniques on the other hand.
- Still in research stage in many areas despite huge efforts by academia and scientists in India as well as abroad

## **ISSUES FOR ENABLING MOBILE WEB IN INDIAN LANGUAGES**

- Character encoding
- Bandwidth and Cost
- Backward Compatibility with Legacy Devices
- Presentation Issues
- Input
- Fonts and Rendering issues
- Lack of standardization
- Input

## Challenges & Issues:

- Development of common messaging framework which is transparent to device, platform and network.
- The UNICODE and UTF8 should be preferred encoding standard for Indian Languages Messaging Standard—though it may require higher bandwidth.
- This will ensure interoperability and proper display of characters for all 22officially recognized languages
- The UNICODE standard should be adopted as this is the most preferred framework for internationalization and Localization
- The backward compatibility issue also need to be addressed.
- The W3C SMIL frame work for mobile multimedia messaging should be adopted as most of the SMIL parameters are embedded in 3GPP/3GPP2 standard
- Any future development of SMS/Multimedia standard needs participation from all stake holders' especially state governments, linguists and language experts.
- ProposedMessagingStandardswouldalsobeusedbysomeoftheSARRCcount ries.
- Therefore,consultations/vettingalsonecessarywiththenationalbodiesofthe secountries.

## The mobileOK Scheme by W3C

MobileOK is designed to improve the Web experience for users of mobile devices by rewarding content providers that adhere to good practice when delivering content to them. MobileOK says nothing about what may be delivered to non-mobile devices; furthermore, mobileOK does not imply endorsement or suitability of content.

For more details please visit : <http://www.w3.org/TR/2009/NOTE-mobileOK-20090825/>

## Mobile ok basic tests

MobileOK Basic Tests 1.0 specifies a number of tests that HTTP responses must pass when a URI is requested with a specific set of HTTP headers in the request. The tests are designed to be machine process able and to provide confidence that content will display well on very basic mobile devices.

For more details please visit: <http://www.w3.org/TR/2008/REC-mobileOK-basic10-tests-20081208/>

## Mobile Web best practices

MobileOK Basic Tests 1.0 is itself based on Mobile Web Best Practices 1.0, which provides a set of sixty guidelines for making content work well across a wide variety of mobile devices.

For more details Please Visit: <http://www.w3.org/TR/2008/REC-mobile-bp-20080729/>

## DDC (Default Delivery Context)

The HTTP Request headers used in mobileOK Basic Tests 1.0 identify a hypothetical user agent called the "Default Delivery Context" (DDC). The values of the key properties of the DDC (screen width, formats supported and other basic characteristics) are set at the minimum possible, while still supporting a Web experience. The DDC is thus not a target to aspire to, it merely sets a base line below which content providers do not need to provide their content. Visit

For more details please visit: <http://www.w3.org/TR/2008/REC-mobile-bp-20080729/#ddc>

## 6. E-Governance Issues

A number of issues in e-governance Application, for example:

- The project implementation is generally vendor driven.
- Lack of standardization (For example, similar projects are carried out by Different state agencies using incompatible file formats and application Standards)
- Reverse compatibility of application with legacy systems are missing in several Projects.
- The IT infrastructures are procured before building the application or digitizing the data.
- Physical security is emphasized, whereas the Logical and application security is left to vendors in many cases.
- Lack of understanding by the departments, for the components of E-governance applications, which can be outsourced or can be carried out in house.
- Some applications are completely in English.
- Some applications have static content in local language but forms in English.
- Script issues in text-to-speech implementations
- Some application are multilingual but only in limited language (e.g. English and only one local language)

## 7.Evaluation

In Evaluation the following condition should be used:

1. **It should Follow W3C Standards:** W3C develops technical specifications and guidelines through a process designed to maximize consensus about the content of a technical report, to ensure high technical and editorial quality, and to earn endorsement by W3C and the broader community.
2. **It should be validated from W3C Validator:** All the Web site of the Government should be validate from W3C Validator.

The following are the W3C Validator:

- CSS Validator
- Html Validator
- Mobile OK Validator
- XHTML Validator
- WCAG-1.0 Validator

### 7.1 Layout and design consistency

Layout of the website refers to the basic framework and structure of the website. Layout determines where the text content, navigation, graphic images and other elements are placed on the web pages. Design of the website, on the other hand, is about the usage of fonts, colours and images on the pages.

Consistency in layout and design throughout the website makes it easier for the visitors to navigate the site. Additionally, visitors do not get a feeling that they have left the website and landed on another website, when the main elements remain the same. One popular method of consistent web design is to use website templates. Templates are documents that web pages are based on. Templates have some particular patterns in them, which are passed on to the web pages that are created using the template.

## 7.2 Information that support all browser

If you ever developed an international application, you know that dealing with every aspect of text translation, local standards, and localized content can be a nightmare. An internationalized application contains several versions of the same content in various languages or formats. For instance, a webmail interface can offer the same service in several languages; only the interface changes.

All in all, dealing with i18n and l10n means that the application can take care of the following:

- Text translation (interface, assets, and content)
- Standards and formats (dates, amounts, numbers, and so on)
- Localized content (many versions of a given object according to a country)

### Changing the Culture for a User

The user culture can be changed during the browsing session--for instance, when a user decides to switch from the English version to the Hindi version or any other language of India of the application, or when a user logs in to the application and uses the language stored in his preferences. That's why the User class offers getter and setter methods for the user culture.

- Make a site with a clear hierarchy and text links. Every page should be reachable from at least one static text link.
- Offer a site map to your users with links that point to the important parts of your site. If the site map has an extremely large number of links, you may want to break the site map into multiple pages.
- Keep the links on a given page to a reasonable number.
- Create a useful, information-rich site, and write pages that clearly and accurately describe your content
- Think about the words users would type to find your pages, and make sure that your site actually includes those words within it.
- Try to use text instead of images to display important names, content, or links. The Google crawler doesn't recognize text contained in images. If you must use images for textual content,

consider using the "ALT" attribute to include a few words of descriptive text.

- Make sure that your <title> elements and ALT attributes are descriptive and accurate.
- Check for broken links and correct HTML.
- If you decide to use dynamic pages (i.e., the URL contains a "?" character), be aware that not every search engine spider crawls dynamic pages as well as static pages. It helps to keep the parameters short and the number of them few.
- Review our image guidelines for best practices on publishing images

### 7.3 Accessibility Architecture

The purpose of this is:

- To act as a tool for developing and assessing public web services
- To improve the accessibility of public websites for both users and producers.

#### WCAG 2.0 have three levels Checklist:

- Level A
- Level AA
- Level AAA

|             | Level A   | Level AA   | Level AAA  |
|-------------|---|--|--|
| perceivable | 1. Text Alternatives: Provide text alternatives for any non-text content<br>2. Time-based Media: Provide alternatives for time-based media.<br>3. Adaptable: Create content that can be presented in different ways (for example simpler layout) without losing information or structure. | 1. Time-based Media: Provide alternatives for time-based media.<br>2. Distinguishable: Make it easier for users to see and hear content including separating foreground from background. | 1. Time-based Media: Provide alternatives for time-based media.<br>2. Distinguishable: Make it easier for users to see and hear content including separating foreground from background. |

|                |   |   |  |
|----------------|---|---|--|
|                | 4. Distinguishable:<br>Make it easier for users to see and hear content including separating foreground from background.  |   |  |
| Operable       | <p>1. Keyboard Accessible: Make all functionality available from a keyboard.</p> <p>2. Enough Time: Provide users enough time to read and use content.</p> <p>3. Enough Time: Provide users enough time to read and use content.</p> <p>4. Navigable: Provide ways to help users navigate, find content, and determine where they are</p> | 1. Navigable:<br>Provide ways to help users navigate, find content, and determine where they are.   | <p>1. Keyboard Accessible: Make all functionality available from a keyboard.</p> <p>2. Enough Time: Provide users enough time to read and use content.</p> <p>3. Seizures: Do not design content in a way that is known to cause seizures.</p> <p>4. Navigable: Provide ways to help users navigate, find content, and determine where they are.</p> |
| Understandable | <p>1. Readable: Make text content readable and understandable.</p> <p>2. Predictable: Make Web pages appear and operate in predictable ways.</p> <p>3. Input Assistance: Help users avoid and correct mistakes</p>  | <p>1. Readable: Make text content readable and understandable</p> <p>2. Predictable: Make Web pages appear and operate in predictable ways.</p> <p>3. Input Assistance: Help users avoid and correct mistakes</p> | <p>1. Readable: Make text content readable and understandable.</p> <p>2. Predictable: Make Web pages appear and operate in predictable ways.</p> <p>3. Input Assistance: Help users avoid and correct mistakes.</p>  |
| Robust         | 1. Compatible:<br>Maximize compatibility with current and future user agents, including assistive technologies.   |   |  |

## 8. Use Cases

Use Cases could be divided into the “provide, engage, enable” modalities for government on the Web

- Provide: public services on the web, either transactional or information services
- Engage: with citizens and businesses, on government terms or on the citizens terms
- Enable: public sector information re-use.

### Topic Areas for Use Cases

| Code Topic Area  | Relates to    | Corresponding Task Force |
|--|---------------|--------------------------|
| Semantic Interoperability<br>(eg. Judicial)                    | G2G           | TF1                      |
| Persistent URIs  | G2C           | TF1                      |
| Performance Data + Citizen Choice                              | G2C           | TF2                      |
| Data Sharing Policy Expression                                 | G2G, G2C, C2G |                          |
| Digital Preservation + Authenticity + Temporal Degradation     | G2G, G2C      | TF1                      |
| IPR Expression   | G2C, G2B      |                          |
| Identification + Authentication                                | G2C           | TF1                      |
| Data Aggregation   | G2G           | TF3                      |
| Your Web Site is your API<br>(eg. RDFa, sitemaps?)             | G2C, G2B      | TF3                      |
| What Data? How does the government decide?                     | G2B, G2C, C2G | TF2                      |
| Participation in Social Media; what are the rules?             | G2C           | TF2                      |
| Temporal Data<br>Legislation/Legal (Law Reports)<br>Geospatial |               | TF1                      |
| Multi channel delivery<br>(back/front)                         | G2C, G2G      |                          |

\*\* G2G – Government to Government

\*\* G2C – Government to Citizen

\*\* TF – Task Force

## Rationale for a Use Case Template

One of the group's core targets is the identification of specific areas where standardization would have an impact on the capabilities and means of interaction between the different actors (citizens and public administrations, companies, etc.). Therefore, the group should:

- Identify areas where existing standards can be leveraged to improve the state-of-the-art of data integration (in a broad sense) in the context of Public Services.
- Identify opportunities for new standards
- Issue a set of recommendations and best practices

As said before, it is of paramount importance that the standards have an impact on a broad community, propose significant improvement or a completely new way of doing things. Therefore, this group will define a set of use cases that depict specific scenarios where an existing or newly proposed standard can prove its value and limitations. It is expected that, at a later stage, this effort may enable the creation of interoperable, standard software systems and tools that allow for the Seamless Integration of Data in the public sector.

When/if possible, the following information should be captured as well for each use case:

- Impact - in terms of audience or improvement of a series of aspects.
- Required existing standards, and need for new standards.
- Limitations known at this stage.

- Expected problems for implementation or deployment.
- Tools expected to implement these recommendations.

## **Use Case structure**

The Use Case Proposals should be submitted using an agreed upon format. This paragraph presents the current view of the group on this format and is under discussion at this stage.

### **Identifier**

A number we can use to reference the Use Case - just in case the Use Case Name is changed. (e.g. UC-EGIG-SID-001).

### **Name**

The name of the Use Case (e.g. Citizen-to-Citizen Data integration via sms).

### **Problem definition**

Define the problem that the standard aims to solve and proposed benefits.

(E.g. This standard defines a framework for the development of ...).

### **Target population**

Identification or description of the target population of the use case – seen as the community where the use case would have its impact. Should refer to one or many of:

- Citizens. Meaning all citizens with the needed infrastructure. (e.g. a mobile phone or internet connection).
- Public Administrations. The benefit solves a problem or improves the situation for public administrations embracing the proposed standard.

- Specific community. When the use case targets a specific set of citizens
- Companies.

## **Description**

A description of the Use case, in terms of interactions of the different actors involved. It is important to highlight the areas in the use case where the proposed standard would play a significant role.

## **Target software**

In case the standard can / should be deployed into a specific software piece, please identify it here. (E.g. the SMS standard should be implemented in mobile handsets, short message centres and related software – billing, rating, etc.).

## **Identified problems or limitations**

A set of problems or limitations that can be identified at this point. The list should be as comprehensive as possible, trying to foresee:

- Inherent limitations of the proposed interaction model.
- Technical difficulties or need for unavailable technology at this moment.
- possible resistance to change
- deployment challenges

## **Priorization**

A conclusion on whether the use case has the critical mass for recommendation, and a justification (e.g low given its poor reach in terms of target population and dubious technical feasibility).

## 9. Need of W3C Validation services for Indian websites

Validating Web documents is an important step which can dramatically help improving and ensuring their quality and it can save a lot of time and money (read more on why validating matters). Validation is, however, neither a full quality check, nor is it strictly equivalent to checking for conformance to the specification.

### Website Errors

The Web page errors that are generated using Web program are considered to identify the measures for quality of Website design. These errors are further divided into major and minor errors using statistical techniques.

#### 1. Major error:

The major errors directly affect the quality of Web site design and developers must concentrate on this category of errors and these should be eliminated. The major errors include: broken links, document type declaration errors, applet usage errors, server connectivity errors, image load errors, frames tag usage errors and title tag with no keyword errors. The major errors are proportional to the down load time of the Web pages. If major errors are minimized then down load time will be automatically reduced and hence it leads to the better quality.

#### 2. Minor errors:

The minor errors are HTML tag errors and these may cause incorrect display of some components of Web pages. The minor errors include: table tag errors, body tag errors, image tag errors, head tag errors, font tag errors, script tag errors, style tag errors, form tag errors, link tag errors and other tag errors. The developers must be attentive so that Web pages can be properly designed with appropriate HTML tags.

## **Evaluating Qualitative Measures for improved Website Design:**

The errors that are found in Websites' of various lead to the necessity of qualitative measures for effective Website design [8]. The head tag errors (HTE), font tag errors (FoTE) and body tag errors (BTE) identify the problems in the text elements of web page. Thus Text formatting measures are to be evaluated. The image tag error (ITE), body tag errors (BTE) and image load errors related to image identifies the errors in display of images and hence Graphic element measures to be evaluated. The table tag errors (TTE), frame tag errors (FTE), style tag errors (StTE), font tag errors (FoTE), frame tag usage errors and document type declaration errors cause the invention of page formatting measures. Link Tag Errors (LTE) and broken links identify the need of link formatting measures. The form tag errors (FmTE), script tag errors (STE) and title tag with no keyword errors identify the need of page performance measure. The script tag errors (STE) applet usage errors, server connectivity errors and broken link errors contribute the need of Website architecture measure.

### **9.1 CSS Validator**

#### **What is CSS Validator and why we need it?**

The W3C CSS Validation Service is free software created by the W3C to help Web designers and Web developers to check Cascading Style Sheets (CSS). It can be used on this free service on the web, or downloaded and used either as a java program, or as a java servlet on a Web server.

If you are a Web developer or a Web designer, this tool will be an invaluable ally. Not only will it compare your style sheets to the CSS specifications, helping you find errors, typos, or incorrect uses of CSS, it will also tell you when your CSS poses some risks in terms of usability.

#### **What does “Valid CSS” mean? Which version of CSS does this Validator use?**

According to the CSS 2.1 Specification: The validity of a style sheet depends on the level of CSS used for the style sheet. [...] valid CSS 2.1 style sheet must be written according to the grammar of CSS 2.1. Furthermore, it must contain only at-rules, property names, and property values defined in this specification

By default, this Validator checks style sheets against the grammar, properties and values defined in the CSS 2.1 specification, but other CSS profiles can be

checked against by using the options. CSS is an evolving language, and it is considered by many that “CSS” is a single grammar (the one defined in the latest specification) with a number of properties and acceptable values defined in various profiles. In a future version of this Validator, the default behaviour may be to check style sheets against that latest “CSS grammar” and the cloud of all standardized CSS properties and values.

## 9.2 XHTML Validator

The Mark up Validator is a free service by W3C that helps check the validity of Web documents.

Most Web documents are written using mark up languages, such as HTML or XHTML. These languages are defined by technical specifications, which usually include a machine-readable formal grammar (and vocabulary). The act of checking a document against these constraints is called validation, and this is what the Mark up Validator does.

Validating Web documents is an important step which can dramatically help improving and ensuring their quality and it can save a lot of time and money (read more on why validating matters). Validation is, however, neither a full quality check, nor is it strictly equivalent to checking for conformance to the specification.

This Validator can process documents written in most mark up languages. Supported document types include the HTML (through HTML 4.01) and XHTML (1.0 and 1.1) family, MathML, SMIL and SVG (1.0 and 1.1, including the mobile profiles). The Mark up Validator can also validate Web documents written with an SGML or XML DTD, provided they use a proper document type declaration.

This Validator is also An HTML validating system conforming to International Standard ISO/IEC 15445—HyperText Mark up Language, and International Standard ISO 8879—Standard Generalized Mark up Language (SGML) – which basically means that in addition to W3C recommendations, it can validate according to these ISO standards.

Before an XHTML file can be validated, a correct DTD must be added as the first line of the file.

The Strict DTD includes elements and attributes that have not been deprecated or do not appear in framesets:

```
!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//HI"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"
```

The Transitional DTD includes everything in the strict DTD plus deprecated elements and attributes:

```
!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//HI"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"
```

The Frameset DTD includes everything in the transitional DTD plus frames as well:

```
!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//HI"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd"
```

### 9.3 Mobile OK Validator

The W3C mobileOK Checker is a free service by W3C that helps check the level of mobile-friendliness of Web documents, and in particular assert whether a Web document is mobileOK.

To understand why checking a Web document for mobile-friendliness really matters, it is probably worth emphasizing a few points about the so-called mobile world. Compared to a regular desktop computer, a mobile device may be regarded as limited at first glance: smaller screen size, smaller processing powers, smaller amount of memory, no mouse, and so on. Compared to fixed data connections, mobile networks can be slow and often have a higher latency. Compared to a user sitting in front of his computer, the user on the go has limited time and is easily distracted. On top of these constraints, the mobile world is highly fragmented: many different devices, each of them defining a unique set of supported features.

For these reasons, although most mobile devices may render Web documents, the user experience when browsing the Web on a mobile device is often poor when a Web document hasn't been designed with mobility in mind.

To help ensure that an appropriate user experience is possible on as many mobile devices as possible, the Mobile Web Best Practices Working Group, part of the W3C Mobile Web Initiative, defined a set of recommended guidelines to follow when creating Web documents: the Mobile Web Best Practices 1.0 specification.

Out of these best practices, the working group extracted a consistent set of best practices that may be automatically checked. This set of best practices defines the notion of mobile-friendliness mentioned above, and this is what the W3C mobileOK Checker actually tests. When a Web document passes all the tests, it is mobileOK. More precisely, the tests run by the W3C mobileOK Checker are formally defined in the mobileOK Basic Tests 1.0 specification.

Being mobileOK is neither a guarantee that the Web document may be rendered correctly by all mobile devices, nor insurance that the user experience was correctly addressed. Further quality improvements based on the full set of the Mobile Web Best Practices may be in order. The Mobile Web Best Practices Flip cards are a useful and handy companion for creators of Web content to keep the Mobile Web Best Practices in mind.



## 10. Important W3C Standards to be adopted

W3C develops technical specifications and guidelines through a process designed to maximize consensus about the content of a technical report, to ensure high technical and editorial quality.

- Gather information about the areas where best practice guidelines are needed: best practices will be drawn from the successes (and failures) of efforts at opening, sharing, and re-using knowledge about the use of standards and specifications by government applications that could be collected into a set of best practices with the intent of identifying productive technical paths toward better public services.
- Provide input on how to ease standards compliance: use previous successful experiences in terms of broad government use (such as the Web Accessibility Initiative work) to identify ways for standard bodies to better speak in terms of government needs; for example, additional effort to package, promote, and train on best practices and existing material and tools.

### Future W3C Standard:

1. Semantic Web
2. CSS Mobile
3. SOAP
4. SPARQL
5. Web Architecture
6. XHTML 2
7. RDF
8. Web Fonts

**Patron:**

**N. Ravi Shanker**

Additional Secretary

**Contact:**

**Ms. Swaran Lata**

Country Manager, W3C India

**Dr. Somnath Chandra**

Deputy Country Manager, W3C India

Department of  
Information Technology  
6, CGO Complex  
Electronics Niketan  
Lodhi Road, New Delhi  
Telephone: +91-011-24364744, 24363525  
**Email:** [swaran@w3.org](mailto:swaran@w3.org), [somnath@w3.org](mailto:somnath@w3.org)