



## SaudiNIC's Best Practices

# Supporting and Managing Arabic Domain Names

August 2017, Ver. 1.0

The intention of this best practice document is to share SaudiNIC's experiences and practices in supporting and managing Arabic domain names.



## Table of Contents

1	Introduction.....	3
2	Major Developments Related to ADNs.....	3
2.1.	Technical Standard for IDNA.....	3
2.2.	Defining Accepted Arabic Letters in ADNs.....	4
2.3.	GCC Pilot Project for Arabic Domain Names.....	4
2.4.	Arab Domain Name Pilot Project.....	4
2.5.	ICANN IDN ccTLD FastTrack.....	5
2.6.	First Non-Latin Domain Names Go Online.....	6
2.7.	Beyond Arabic Domain Names.....	6
2.8	List of Existing Arabic IDN TLDs.....	7
3.	Concepts and Practices for Supporting ADNs.....	8
3.1.	Concepts.....	8
3.1.1.	Language Table.....	8
3.1.2.	Variant Table.....	9
3.1.3.	Step-by-Step Language Support.....	10
3.1.4.	Variants relationship Types and Actions.....	11
3.1.5.	Study Variants Across the Whole Arabic Script.....	15
3.1.6.	Variants Base on Character Positions.....	15
3.1.7.	A Label is Composed Using a Single Input Device.....	17
3.1.8.	Must be Allocated Variants.....	19
3.1.9.	One Key for All Variants (Master key).....	20
3.1.10.	Simple User Interface.....	21
3.1.11.	Variant Filters.....	21
3.1.12.	Consistent Variants.....	25
3.2.	Practices.....	26
3.2.1.	Identifying Supported Language(s).....	26
3.2.2.	Acquiring Language and Variant Tables.....	27
3.2.3.	Resolving Variant Conflicts and Finalizing Variant Tables.....	33



3.2.4. Implement Variant Keys .....	33
3.2.5. Provide clever tools to manage variants (VMS).....	34
4. Other Considerations.....	36
4.1. Regulation Issues.....	36
4.2. Reserved List Issues.....	37
4.3 Technical issues .....	37
Appendix 1 (Language tables) :.....	38
I. Arabic Language table.....	38
II. Urdu language table .....	39
III. Persian language table.....	40
IV. Malay (Jawi) language table .....	41
V. Pashto language table.....	42
Appendix 2: Arabic language table in IANA new XML format .....	43
Appendix 3: Guideline Rules for writing Arabic Labels under (السعودية) .....	57
Appendix 4: Control Characters: ZWJ and ZWNJ.....	60



# Supporting and Managing Arabic Domain Names

## SaudiNIC's Best Practices

### 1. Introduction

The goal of this document is to outline and summarize some best practices and lessons learned in providing registration services and managing Arabic domain names (ADN). The prime intended audiences of this document are Arabic IDN (gTLD or ccTLD) registries.

The document is heavily prepared based on SaudiNIC's experiences that expand over a period of time exceeds more than 15 years. SaudiNIC is considered as one of the pioneer supporter for IDN in general and Arabic domain names in particular.

It is worth to mention that based on SaudiNIC's extensive accumulated experiences and practices in this field since 2001, and based on SaudiNIC's continuous efforts to find and reach acceptable, realistic and workable solutions (for all relevant entities: registries, registrants and normal Internet users) to the hassle of managing and using enormous variant domains that might be generated due to character similarities within the whole Arabic script. SaudiNIC has indeed developed and implemented a Variants Management System (VMS) that explicitly and constructively enforces the concepts and practices that will be explained in this document.

### 2. Major Developments Related to ADNs

In the following subsections, brief narrative notes on some of the most important milestones on the road to the realization of Arabic domain name will be presented.

#### 2.1. Technical Standard for IDNA

In 2003, a set of RFCs (RFC 3454, RFC 3490, RFC 3491 and RFC 3492) was published by IETF that enabled using non ASCII characters in domain names under the name IDNA (Internationalized Domain Names in Applications). They provided a technical solution on how IDN could be introduced in a standard way. These RFCs were later updated in 2010 (RFC 5890, RFC 5891, RFC 5892, RFC 5893 and RFC 5894).



### 2.2. Defining Accepted Arabic Letters in ADNs

A number of expert groups were formed under the supervision of intergovernmental organizations or interest consortiums, such as Arab League, ESCWA, and AINC. In 2003, a group of experts in the fields of Arabic language and domain names gathered to discuss issues related to Arabic domain names (ADNs) such as: the accepted characters that should be used in writing ADNs and the proposed Arabic TLDs for the Arab countries. Then the draft was reviewed by the Arabic Team for Domain Names and submitted to the Arab League for final approval. The main parts of the document were the accepted Arabic character set and the Arabic Top-Level domains. That effort ended later by issuing an Internet draft that specified the set of allowed characters in ADNs along with other linguistic issues. That document evolved later to become RFC 5564. It was agreed by the team to follow this document by members of the Arab League, leaving sufficient space for each IDN ccTLD manager to draft their own guidelines for registering ADNs that should not conflict with the published RFC.

### 2.3. GCC Pilot Project for Arabic Domain Names

Before the launching of IDN ccTLDs FastTrack by ICANN, the managers of the GCC (Gulf Cooperation Council) ccTLDs (i.e., .ae, .bh, .kw, .om, .qa, and .sa) in their meeting on 7th of March 2004 agreed to initiate a pilot project for Arabic domain names. The goal of the project was to implement a testbed environment for ADNs. The project allowed all GCC countries to early experience the use of ADNs, identified the needs, located possible problems, and developed some tools. The main objectives of the project were:

- To gain experience and knowledge in supporting ADNs and share it with Arab countries.
- Test the implantations of ADNs.
- Build the local awareness about ADNs.
- Establish joint work with other entities (i.e., ISPs, universities).
- Develop some tools related to ADNs and DNS.

The project team was assembled from one or two technical persons from each GCC ccTLD ( ae, bh, kw, om, qa, sa). By the end of the project a number of important achievements were realized, including the success in preparing a GCC root servers and test GCC ADNs. Also, some important technical guidelines were drafted as a result of this project along with policies and regulations for registering ADNs.

### 2.4. Arab Domain Name Pilot Project

The success of the GCC pilot project had led the Arab Team for Domain Names, in their 2nd meeting that was held in Cairo on the 7th and 9th of May 2005, to recommend the expansion of the GCC Pilot Project for Arabic Domain Names to include all members of the Arab League (22 countries). Hence, the project was renamed as follows: **Arabic Domain Names Pilot Project**, and it became under the auspices of the Arab League. Two committees were created for the management



and operation of the project: a steering committee and a technical committee. The Steering Committee's tasks included the general supervision of the project, management and supervision of the Arabic root servers, and setting policies and procedures which included participation policies and use terms and conditions. While the Technical Committee's tasks included providing technical support for participants and users, technical coordination between participants, technical supervision of the Arabic root servers, and enhancing and improving the project from a technical stand point.

The mission of the project was:

“Implementing a testbed for Arabic domain names (ADN) in the Arab world. This will allow for the early experience the use of Arabic domain names by all Arab countries, the identification of their needs, the agreement upon uniform standards, the identification of possible problems, and the development of required tools and policies.”

The project was expected to contribute to the following strategic objectives:

- Establish and implement Arabic domain names.
- Increase Internet use in the Arab world by addressing linguistic barriers facing Arabic-speaking users.
- Promote the use of Arabic language and to increase the Arabic content on the Internet.
- Promote Arab cultural identity on the Internet.

While the main objectives of the project were:

- Make the Internet easier to use for native Arabic speakers.
- Gain experience and knowledge in the use of Arabic domain names and share it with the Internet community.
- Test the implantations of Arabic domain names based upon the guidelines drafted by the “Arabic Team for Domain Names”.
- Build local awareness related to Arabic domain names.
- Possibly, to develop necessary tools required for Arabic domain names and DNS.
- Develop required policies and guidelines that help achieve the above objectives.

More information about the project can be found at the project website [www.arabic-domains.org](http://www.arabic-domains.org).

### 2.5. ICANN IDN ccTLD FastTrack

The first breakthrough in the history of Arabic domain names and IDN in general, was the ICANN announcement for opening the fast track to apply for IDN ccTLDs on 16-November-2009. Saudi Arabia was among the first countries to apply in the fast track, and it was one of the first countries that their IDN ccTLD to get approved (along with



UAE, Egypt and Russia). Saudi Arabia applied and obtained the following Arabic IDN ccTLD (السعودية) .

### 2.6. First Non-Latin Domain Names Go Online

Thursday 6 May 2010, was a remarkable day in the history of the Internet. In that day, ICANN officially launched IDNs, for the first time in the Internet's history, non-Latin characters could be used for an entire Internet address name. The first (internationalized) domain names (IDNs) were entered into the Internet's root zone on that day, marking a historic first in ICANN's globalization of the Internet.

Saudi Arabia, Egypt, and the United Arab Emirates are the first three countries to use Arabic characters in the last portion of their Internet domain names – that portion to the right of the dot (or to the left for languages like Arabic which written and read from right-to-left).

As soon as the IDN ccTLD for Saudi Arabia (السعودية) was added to the root servers a number of Arabic domain names automatically became among the first Arabic domain names to be on the Internet, e.g.:

- سجل.السعودية xn--rgbn6c.xn--mgbep4a5d4ar
- موقع.السعودية xn--4gbrim.xn--mgbep4a5d4ar
- مركز التسجيل.السعودية xn--mgbggrfi2ikdb7d.xn--mgbep4a5d4ar

Following that, Saudi Arabia was the first country to provide complete Arabic domain name registration services.

### 2.7. Beyond Arabic Domain Names

After the official support of the Arabic domain names registration, users from the local community were able to obtain domain names in their native language. However, utilizing the potentials of the Internet requires the introduction of the services based on Arabic domains, one of them was (Raseel project – an email with Arabic addresses). The goal of that project was to develop a web based Arabic Email system that allow its users to obtain Arabic Email addresses and then use them to communicate in the same way as the regular email .

The project team successfully completed all the technical requirements of the project and lunched the online web email system supporting fully Arabic domain names, also a dedicated website was lunched (رسيل.السعودية) to be the official reference of the project and the main communication getaway with the users. The project team managed also to successfully develop a plug-in that was used to correctly display and handle Arabic domain names in the popular mail client (MS Outlook – versions 2007 and 2010). In addition, a focused grope of selected participants from CITC, KACST and other national and international entities were invited to test the system and the plug-in and provide their feedback.



## 2.8. List of Existing Arabic IDN TLDs

The following table lists the existing (as of 15 Aug 2017) Arabic script IDN TLDs. There are 34 TLDs, 13 of them are IDN gTLDs.

IDN TLD	Type	Registry
.ابوظبي	gTLD	Abu Dhabi Systems and Information Centre
.اتصالات	gTLD	Emirates Telecommunications Corporation (trading as Etisalat)
.إختبار	Test	Internet Assigned Numbers Authority
.ارامكو	gTLD	Aramco Services Company
.آزمایشی	Test	Internet Assigned Numbers Authority
.الأردن	ccTLD	National Information Technology Center (NITC)
.الجزائر	ccTLD	CERIST
.السعودية	ccTLD	Communications and Information Technology Commission
.العليان	gTLD	Crescent Holding GmbH
.المغرب	ccTLD	Agence Nationale de Réglementation des Télécommunications (ANRT)
.امارات	ccTLD	Telecommunications Regulatory Authority (TRA)
.ایران	ccTLD	Institute for Research in Fundamental Sciences (IPM)
.یارت	ccTLD	National Internet eXchange of India
.بازار	gTLD	CORE Association
.بھارت	ccTLD	National Internet Exchange of India
.بيتك	gTLD	Kuwait Finance House
.پارت	ccTLD	National Internet eXchange of India
.پاکستان	ccTLD	National Telecommunication Corporation
.تونس	ccTLD	Agence Tunisienne d'Internet
.سودان	ccTLD	Sudan Internet Society
.سورية	ccTLD	National Agency for Network Services (NANS)
.شبكة	gTLD	International Domain Registry Pty. Ltd.
.عراق	ccTLD	Communications and Media Commission (CMC)
.عرب	gTLD	League of Arab States
.عمان	ccTLD	Telecommunications Regulatory Authority (TRA)
.فلسطين	ccTLD	Ministry of Telecom & Information Technology (MTIT)
.قطر	ccTLD	Communications Regulatory Authority
.كاتوليك	gTLD	Pontificium Consilium de Communicationibus Socialibus ((PCCS) (Pontifical Council for Social Communication)
.كوم	gTLD	VeriSign Sarl
.مصر	ccTLD	National Telecommunication Regulatory Authority - NTRA
.مليسيا	ccTLD	MYNIC Berhad
.موبايلى	gTLD	GreenTech Consultancy Company W.L.L.
.موقع	gTLD	Suhub Electronic Establishment
.همراه	gTLD	Asia Green IT System Bilgisayar San. ve Tic. Ltd. Sti.

Source: <https://www.iana.org/domains/root/db>



## 3. Concepts and Practices for Supporting ADNs

The Arabic script is the 2nd most widely used alphabetic writing system in the world (more than 43 countries) and is used by many languages such as: Arabic, Urdu, Persian, Turkish, Kurdish, Pashto, Malay, ... etc. Therefore, it is expected that more than one billion potential users could be concerned in using Arabic script domain names.

The Arabic script brings up many challenges for IDN TLD registries. For example, the noticeable and the foremost challenge is how to handle and manage character similarities (i.e., variants). The Arabic script block of the UNICODE table consists of a number of groups of characters that have the same shapes (Homoglyph), e.g., Kaf, Heh, Yeh, Alef, ... groups.

This necessitate that Arabic script IDN registries to follow certain practices to complete and finalize some basic prerequisite requirements. The fundamental objective is to secure the IDN TLD name space in a simple and reasonable approach by:

- Enhancing security through preventing or reducing domain name phishing.
- Ensuring domain name reachability world-wide.
- Simplifying the use and management of IDNs and their variants.

Working with an evolving topic and developing new solutions necessitate using, defining and making up some concepts.

The following subsections will highlight in a very general manner the major concepts used in supporting and managing IDN variants. These concepts are being followed and implemented by SaudiNIC in their practices to achieve the necessary requirements. Later subsections will outline the fundamental practices to complete the necessary requirements in supporting and managing IDN variants by registries. They will include some summaries of SaudiNIC's current practices.

### 3.1. Concepts

Here is a list of major concepts with their brief explanations that have been implemented or followed by SaudiNIC in their implementation of its Variants Management System (VMS) towards supporting and managing Arabic domain names.

#### 3.1.1. Language Table

A language table represents a list of permitted code points (letters, digits, and symbols) that are allowed for writing a domain name in a specific language. IANA maintains a collection of "IDN tables" ([Repository of IDN Practices](#)) which represent permitted code points allowed for IDN registrations in particular registries.



Basically, code points of a language table should cover the alphabetic of that language and should adhere to the Letter–Digit–Hyphen (LDH) concept that was applied to the existing ASCII domains. Language tables should be prepared with the help of members of language communities or language experts.

It is worth to mention that domain names are technical labels to assist in accessing Internet resources. Therefore, they are not intended to be totally grammatically correct or free of pronunciation errors. Moreover, they are not intended for writing poems or meaningful sentences. Nevertheless, there should be a balance between the language needs and the ease of use. Therefore, it must be sufficient to include main letters and digits that are agreeable by the users of the language. Hence, the approved language table should not turn away users from registering and using IDN domains because of the complexity or language requirements (spelling or grammatical). Additionally, supported code points listed in the table should be all part of a standard keyboard and supported by common operating systems.

### **Benefits of the Concept** (Language Table):

- Language tables are very powerful tools to:
  - control input via the user interface (so no mixing between code points that does not belong to a language).
  - help identifying “[must be allocated variants](#)” for reachability purposes.
  - tremendously reduce the number of unnecessary allocatable variants.
  - protect the TLD-space by defining minimum set of supported languages (white-list inclusion method).
  - easily help adding a new supported language by the TLD.

### 3.1.2. *Variant Table*

A variant table is a table that captures relationships between code points from a language table and any other code points within the same language table or the whole Arabic script. It should be generated with the help of members of the language community and/or language experts.

Two code points are considered variants if:

- i. They are visually identical in any one of the four character positions (isolated, initial, medial or final). This includes also, if they are visually similar and they are used as stylistic variations of one another within the calligraphic tradition.
- ii. They are visually similar to the degree that they may be considered as stylistic variations of each other, even where such consideration is not part of the established calligraphic tradition.



- iii. They only differ in their dot orientation, i.e. where those dots, which are used as part of the letter, are rotated but their count and placement (above or below) does not change.

When identifying variants and constructing a variant table, the following should be considered:

1. Study the similarities (variants) for the supported language code points in the following perspectives:
  - a. Defining variants within the language itself (i.e., studying variant relationships within the code points of the language table, e.g. ١ & ا and ٣ & ٤ ...etc.).
  - b. Defining variants across the whole script (i.e., studying variant relationships of all code points of the language table against the remaining code points within the Arabic script, e.g. Arabic KAF & Urdu KAF). For more information see the Section "[across the whole script](#)".
2. Finalize [variant Types/Action](#), e.g. Exact, Typo, Allocated, Blocked, etc.
3. Defining variants that are needed for international reachability across different input devices. Thus, the variant table should determine "[must be allocated variants](#)", i.e., identify how users may type a domain name using different input devices from other languages. For example, Arabic user may use Urdu keyboard to register and/or reach an Arabic domain name.

### **Benefits of the Concept** (Variant Table):

- Securing the registry domain space against phishing attacks (grouping all variants under one key).
- Defining what are the needed variants for a language community.
- Defining what are the needed variants for other language community within the Arabic script.

### **3.1.3. Step-by-Step Language Support**

As stated before, the Arabic script is being used as a writing system by many languages. However, a large group of them either they aren't used (i.e., historical languages or have switched to different scripts), or they aren't digitally mature enough from linguistic and technical point of view (e.g., no electronic existence). A language is considered to be mature (in this context) whenever it has:

- *a language and variant tables available (published by the relevant language community or registry).*
- *a well-known input device (keyboard layout) that support all code points listed in the language table and being adopted by well-known OS providers.*
- *some online contents.*

Therefore, a variant management system should handle this issue straightforwardly by incorporating "mature languages" first in its system for immediate support and



protection to both the TLD name space and registrants. Later at any time, whenever another language gets ready (by having its language and variant tables being defined) then it can be easily added to the registry system (i.e., no need to regenerate the variant keys for the registered domain names unless for rare cases where there are some conflicts that need to be resolved before adding the new language).

#### **Benefits of the Concept** (Step-by-Step Language Support):

- Quick start to ready languages
- Flexibility to add more language as they become ready
- Keeping the registry domain space safe until things are clear.

#### 3.1.4. *Variants relationship Types and Actions*

When studying and creating variant tables, variants should be classified based on their types:

- **Exact:** this means that the relationship similarity between the concerned characters is visually identical (i.e., as mirror).
- **Typo:** this means that the relationship similarity between the concerned characters is considered somehow look-alike but not identical (typo/style match or interchangeable characters).

In the following example, the code points (0643, 06A9, and 06AA) have the “Exact” relationships in the positions indicated by the green background. Other position are considered as “Typo” relationship (yellow background).

Code Points	Possible shapes in context [isolated, final, medial, initial]			
 0643 LETTER KAF				
 06A9 LETTER KEHEH				
 06AA LETTER SWASH KAF				



In the following example, the code points (0649, 064A, and 06CC) have the “Exact” relationships in the positions indicated by the green background. Other position are considered as “Typo” relationship (yellow background).

Code Points	Possible shapes in context [isolated, final, medial, initial]			
 0649 LETTER ALEF MAKSURA				
 064A LETTER YE H				
 06CC LETTER FARSI YE H				

It is expected that the “Exact” relationships will be almost the same across languages in the Arabic script. However, the “Typo” relationships may differ depending on the language community expectation. The “Typo” relationship will help later in solving conflicts when merging multiple languages variant tables together.

The “Typo” relationship includes the following cases:

- Look alike characters
- Characters that are used interchangeably (e.g., In the Arabic language, at the end of words, ARABIC LETTER TEH MARBUTA (U+0629) and ARABIC LETTER HEH (U+0647) are used interchangeably in writing. That is because they sound similar when pronounced at the end of a phrase, and hence the LETTER TEH MARBUTA sometimes is written as LETTER HEH and the two are considered "confusable" in that context. See RFC 6365.).
- Different dot orientation.



Variant classifications based on their type (Exact or Typo) will assist in the process of merging language table and resolving conflicts particularly when the relationship is typo.

Additionally, when studying and creating variant tables, action related to each variants should be determined as follows:

- **Blocked:** this will designate any generated variants that consist of the corresponding code point as Blocked. Blocked variants can't be activated (enabled and used) by the registrant. Furthermore, they will not be available for registration to others. This action is used mainly for protecting the TLD name space as well as protecting registrants' domain names.
- **Allocate-able:** this action will designate any generated variants that consist of the corresponding code point as Allocate-able. Allocate-able variants are not available for registration to others while they can be activated (enabled and used) by the registrant only when needed. Allocate-able variants are further classified into three sub-groups:
  - **Must-be-allocated:** these are small set of variants that must be allocated for reachability purposes (see the Section [Must be Allocated Variants](#)). Hence, it is recommended to enable them since they are needed for domain name stability and reachability.
  - **Desired:** a short list of allocate-able variants that are designated using some filters for the purpose to simplify displaying, managing and selecting variants by registrants (see the Section [Variant Filters](#) ).
  - **Undesired** the remaining list of allocate-able variants that are not designated as Must-be-allocated or Desired variants. Mostly they consists of variants that are not attractive or practical to enable (see the Section [Variant Filters](#) ).

Let's consider the following example for the label (مكة-المكرمة) that was generated by the [SaudiNIC's Variants Management System](#)

**Statistics Summary:**

Total Variants	3239
I. Must be Allocated Variants	2
II. Desired Variants	4
III. Not desired Variants	28
IV. Blocked Variants	3205

**I. Input:**

LANGUAGE	UNICODE	LABEL
Arabic	(U+0645) (U+0643) (U+0629) (U+002D) (U+0627) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0629)	مكة-المكرمة

**II. Must be Allocated Variants (2):**

LANGUAGE	UNICODE	LABEL
Persian, Malay, Pashto	(U+0645) (U+06A9) (U+0629) (U+002D) (U+0627) (U+0644) (U+0645) (U+06A9) (U+0631) (U+0645) (U+0629)	مكة-المكرمة
Urdu	(U+0645) (U+06A9) (U+06C3) (U+002D) (U+0627) (U+0644) (U+0645) (U+06A9) (U+0631) (U+0645) (U+06C3)	مكة-المكرمة

**III. Desired Variants (4):**

UNICODE	LABEL
(U+0645) (U+0643) (U+0647) (U+002D) (U+0627) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0647)	مكة-المكرمه
(U+0645) (U+06A9) (U+0647) (U+002D) (U+0627) (U+0644) (U+0645) (U+06A9) (U+0631) (U+0645) (U+0647)	مكة-المكرمه
(U+0645) (U+0643) (U+0629) (U+002D) (U+0627) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0647)	مكة-المكرمه
(U+0645) (U+0643) (U+0647) (U+002D) (U+0627) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0629)	مكة-المكرمة

**IV. Not Desired Variants (28) ... here is a sample**

UNICODE	LABEL
(U+0645) (U+0643) (U+0629) (U+002D) (U+0622) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0629)	مكة-المكرمة
(U+0645) (U+0643) (U+0647) (U+002D) (U+0622) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0629)	مكة-المكرمة
(U+0645) (U+0643) (U+0629) (U+002D) (U+0623) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0629)	مكة-المكرمة
(U+0645) (U+0643) (U+0647) (U+002D) (U+0623) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0629)	مكة-المكرمة
(U+0645) (U+0643) (U+0629) (U+002D) (U+0625) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0629)	مكة-المكرمة
(U+0645) (U+0643) (U+0647) (U+002D) (U+0625) (U+0644) (U+0645) (U+0643) (U+0631) (U+0645) (U+0629)	مكة-المكرمة
(U+0645) (U+06A9) (U+0647) (U+002D) (U+0627) (U+0644) (U+0645) (U+06A9) (U+0631) (U+0645) (U+0629)	مكة-المكرمة

Please note as we have developed our work before creating and publishing the relevant RFCs (e.g., RFC 7940, RFC 3743), we have used a set of terminologies that might appear different from the one used in these RFCs. However, they are somehow representing the same things. The following table lists our terminologies with the corresponding RFC's terminologies.



SaudiNIC's terminologies	RFC's terminologies
Blocked	Blocked
allocate-able	allocateable
desired allocate-able	allocateable
undesired allocate-able	allocateable
must-be-allocated	activated (or "allocateable" based on the registry choice)
activate	activated

**Benefits of the Concept** (Variants relationship Types and Actions):

- Help in identifying the correct variants for a domain name
- Help in identifying the proper actions
- Help the registry to list the suitable variants for registrants
- Help in protecting the domain names space
- Help both the registrant and the user to reach a domain name

### 3.1.5. Study Variants Across the Whole Arabic Script

When building a variant table, it is highly recommended, for each code point in the supported language table, to conduct a full study across the whole Arabic Script in order to identify all possible variants corresponding to the code point. It is not enough to only check variants within code points of the supported languages.

Studying variants across the whole Arabic script allows adding a new supported language to the variant management system without the need to re-study all the previous supported languages and change their variant tables. Thus, the variant management system archives better efficiency by eliminating the need to re-study all languages again and less efforts in key regeneration when adding new languages to the registry later on.

**Benefits of the Concept** (Study Variants Across the Whole Arabic Script):

- Protection to the registry name space regardless of the supported languages
- Doing the work for a language one time
- Flexibility to add more language as they become ready without effecting existing supported language
- No need to re-generate variant keys when new language is added

### 3.1.6. Variants Base on Character Positions

In Arabic script languages, a character (code point) may take different shapes depending on its position (standalone, beginning, middle, or end) within a word.



Therefore, **character positions** ought to be considered when deciding whether two code points are variant or not.

For illustration purposes for this concept, let's consider the following variant table (HEH Class).

Code Point	Possible shapes in context [Isolated, Final, medial, Initial]			
0647	ه	هـ	هـ	هـ
06BE	هـ	هـ	هـ	هـ
06C1	ه	هـ	هـ	هـ
06D5	ه	n/a	n/a	n/a

Based on this table, the following is a list of all the permutations (total 16) of the Arabic word (هدهد) (meaning in English the bird Hoopoe):

هدهد	هدهد	هدهد	هدهد
هدهد	هدهد	هدهد	هدهد
هدهد	هدهد	هدهد	هدهد
هدهد	هدهد	هدهد	هدهد

If we don't consider character positions when generating variants then we would get a list of (16) variants. However, if we consider character positions when generating variants, we will get only (4) valid variants (i.e. 25%). So the other invalid 12 (75%) words will NOT be considered variants as they are totally different and don't present risk or any security issue (more accuracy).

**Benefits of the Concept** (Variants Base on Character Positions):

- Reduce the number of generated variants.
- Precise variant accuracy



Please note that it is all right to sacrifice “precise variant accuracy“ to have a simpler variant management system by not employing this concept. Hence, this concept (Variants Base on Character Positions) is not necessary to implement.

### 3.1.7. A Label is Composed Using a Single Input Device

From practical and realistic point of view, it is safe to assume that a string (label) in Arabic script based language is typed using “one” input device (keyboard); i.e., there are no mixing between code points from different keyboards (i.e., **no language mixing**).

For example, when typing the word (كريم), the possible valid ways that can be typed subject to the selected input device (keyboard) will be the following (total 3) (assuming 5 languages only are supported):

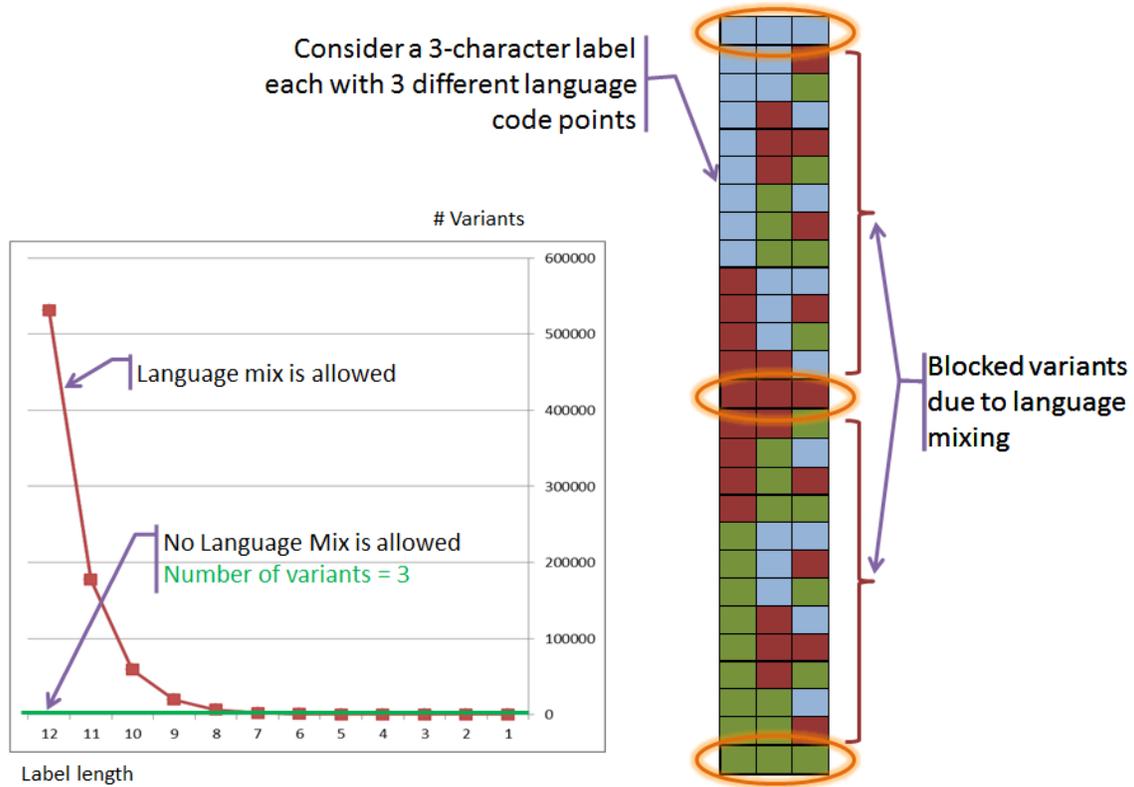
LANGUAGE	UNICODE	LABEL
Arabic	(U+0643) (U+0631) (U+064A) (U+0645)	كريم
Malay,	(U+06A9) (U+0631) (U+064A) (U+0645)	كريم
Persian, Urdu, Pashto	(U+06A9) (U+0631) (U+06CC) (U+0645)	كريم

Other possible combinations such as the following:

(U+06A9) (U+0631) (U+06D0) (U+0645)	كريم
(U+0643) (U+0631) (U+06CC) (U+0645)	كريم
(U+06AA) (U+0631) (U+064A) (U+0645)	كريم
(U+06AA) (U+0631) (U+06CC) (U+0645)	كريم
(U+0643) (U+0631) (U+06D0) (U+0645)	كريم
(U+06AA) (U+0631) (U+06D0) (U+0645)	كريم
(U+0643) (U+0631) (U+067B) (U+0645)	كريم
(U+06A9) (U+0631) (U+067B) (U+0645)	كريم
(U+06AA) (U+0631) (U+067B) (U+0645)	كريم

are not realistic nor practical, as each word is composed of characters that are not available in one input device (i.e., you need more than one input device to be able to compose the whole word). Therefore, out of 12 total possible variants for the word (كريم), only three are allocate-able; the rest (which represents %75) are blocked due to language mixing.

For example, consider a 3-character U-label each with 3 different code points derived from the supported languages. Then, probabilistically, there will be 3 to the power of 3 (=27) variants. If it is 4-character label then it will be 3 to the power of 4 (=81), and so on. However, if the concept of “**no language mixing**” is applied, then there will be only 3 variants at the maximum.



The following table further illustrate the strength and efficiency of “**no language mixing**” concept in significantly minimizing the number of allocate-able variants and maximizing the number of blocked variants:

IDN	Total Variants	Allocatable	Blocked	Blocked due to Language Mixing
مكة-المكرمة	3239	34	3205	3181 (99.25%)
القرآن-الكريم	11999	111	11888	11836 (99.56%)
هيئة-الإعلام	47999	81	47918	47764 (99.68%)
كهف-الياسمين	28799	65	28734	28680 (99.81%)
كهف-اكيا	21599	47	21552	21534 (99.92%)

It is clear from this example that the concept “**no language mixing**” noticeably and significantly cuts down the number of allocate-able variants and increases the number of blocked unrealistic variants (blocked variants due to language mixing represent the greater parts of blocked variants where they reach more than 99%). Hence, it definitely adds more accuracy to the number of allocate-able variants.



**Benefits of the Concept** (No Language Mixing):

- Control user input via the interface of the registration and variant management system.
- Help identifying “**must-be-allocated**” variants for reachability purposes.
- Tremendously reduce the number of unnecessary allocate-able variants
- Protect the TLD-space.
- Provide the registrant with a manageable number of variants to choose from, i.e., a better variant management user interface.

**3.1.8. Must be Allocated Variants**

One of the main principles for the stability of the Internet and Internationalized domain names is that the end user should be able to reach a website connected to his/her domain name regardless of location. In order to enforce this principle the input devices (language table) that the user may use to reach a domain name (based on the user location) should be carefully considered when defining variants. Consequently, the variant management system should be aware of supported languages (through language tables) and input devices (through variant tables).

Consider, for instance, the case where a suitable variant is not allocated to the registrant this may cause a reachability problem and reduce the user acceptance. For example, if someone registered the domain name “مكة” (all characters from the Arabic language) and a user try to reach the website connected to this domain name from an Internet café, say, in Pakistan. He/she will not be able to reach that website unless if the variant “مكة” (Urdu variant) is already allocated and activated.

LANGUAGE	UNICODE	LABEL
Arabic	(U+0645)	مكة
	(U+0643)	
	(U+0629)	
Persian, Malay, Pashto	(U+0645)	مكة
	(U+06A9)	
	(U+0629)	
Urdu	(U+0645)	مكة
	(U+06A9)	
	(U+06C3)	

**Must be allocated variants for reachability purposes (for community using different input devices)**



Thus, variants need to be studied from both similarity point of view (by language community) and reachability point of view (based on input devices used by other language communities). Therefore, for reachability purposes, it is believed that variants which are generated by the latter “**must be allocated**” (automatically) to the registrant since they are needed for domain name stability and reachability. So that, a registered domain name is accessed regardless of the input devices (language table) being used by the navigator users.

**Benefits of the Concept (Must be Allocated):**

- Enhance reachability regardless of input device
- Enhance both user and universal acceptance

**3.1.9. One Key for All Variants (Master key)**

Due to its characteristics, a domain name label written in an Arabic script is expected to have multiple variants. In many cases, the total number of variants may reach thousands and millions depending on the characters forming the label and its length. Here are some examples for different Arabic words and how the number of variants dramatically grow as the domain name get longer:

Label	Approximately # of variants
اتصال	300
اتصالات	6,000
الاتصالات	60,000
هيئة-الاتصالات	2,879,999
هيئة-الاتصالات-وتقنية-المعلومات	82,944,000,000

Therefore, generating, displaying or storing all possible variants are not a visible nor a practical solution, especially for longer domain names as they generate huge variant lists.

Thus, an identification mechanism should be developed and used to easily manage the whole variants set with one unique identifier. This will speed up the lookup process for both domain name availability and variant allocation (process efficiency) and eliminate the need for saving all possible variants (storage efficiency).

SaudiNIC has developed a new algorithm used in its Variant Management System called “[Master Key Algorithm](#)” to generate a unique key for a domain name label and all of its possible variants.



### **Benefits of the Concept** (Master Key):

- Easily manage the whole variants set with one unique identifier.
- Speed up the lookup process for both domain name availability and variant allocation.
- Eliminate the need for saving all possible variants and hence save disk space.

### *3.1.10. Simple User Interface*

Normal registrants may not be aware of concepts and terminologies related to variants such as code points, similarly confusing characters, blocked, allocated, must be allocated, allocate-able variants, etc.

Also, registrants would have difficulty to generate all variants, differentiate between all of them, determine which one would be blocked automatically, determine which one must be allocated for reachability, determine which variants would be suitable (optionally allocated), how to enable variants, and are there some variants activated automatically?

Additionally, The registry should not assume that regular user will be able to know the differences between KAF (U+0643) and KEHEH (U+06A9) just by displaying the different variant labels. Also, it is unpractical to list all variants or just allocate-able variants as they may exceed hundreds of allocate-able variants.

Thus, a simple registration and variant management systems should be developed to help registrants to manage and enable desired variants either through manually typing them or selecting from a displayed (desired) list. For example, the Registry could provide a separate web interface (and/or EPP command) for listing the possible variants in a clever way (i.e., desired allocate-able variants) to help both the Registrar and Registrant generating and managing variants.

### **Benefits of the Concept** (Simple User Interface):

- Help the user to choose the suitable and need variants.
- Hopefully, help hosting companies to automatically host all variants.
- Enhance both user and universal acceptance.

### *3.1.11. Variant Filters*

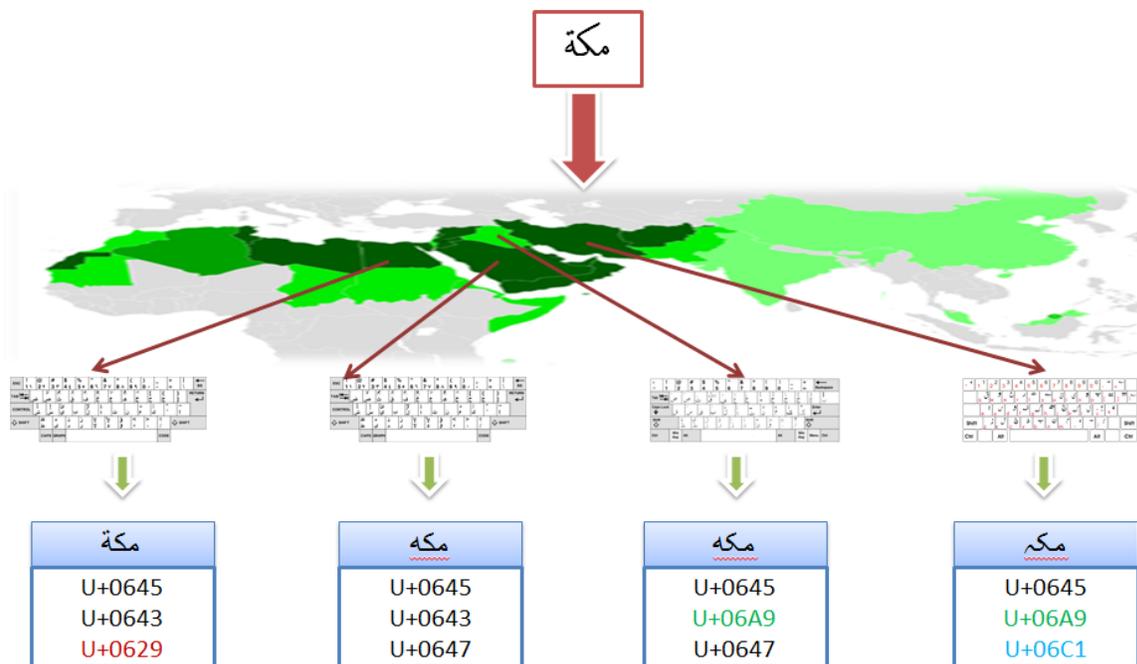
Due to the characteristics of the Arabic script and languages, it is normal to have huge set of variants to a label particularly as the label length increases. Variants are generated due to:

- The UNICODE table for the Arabic script block contains a number of groups of characters that have the same shapes (Homoglyph): e.g., Kaf, Heh, Yeh, Alef, ... groups

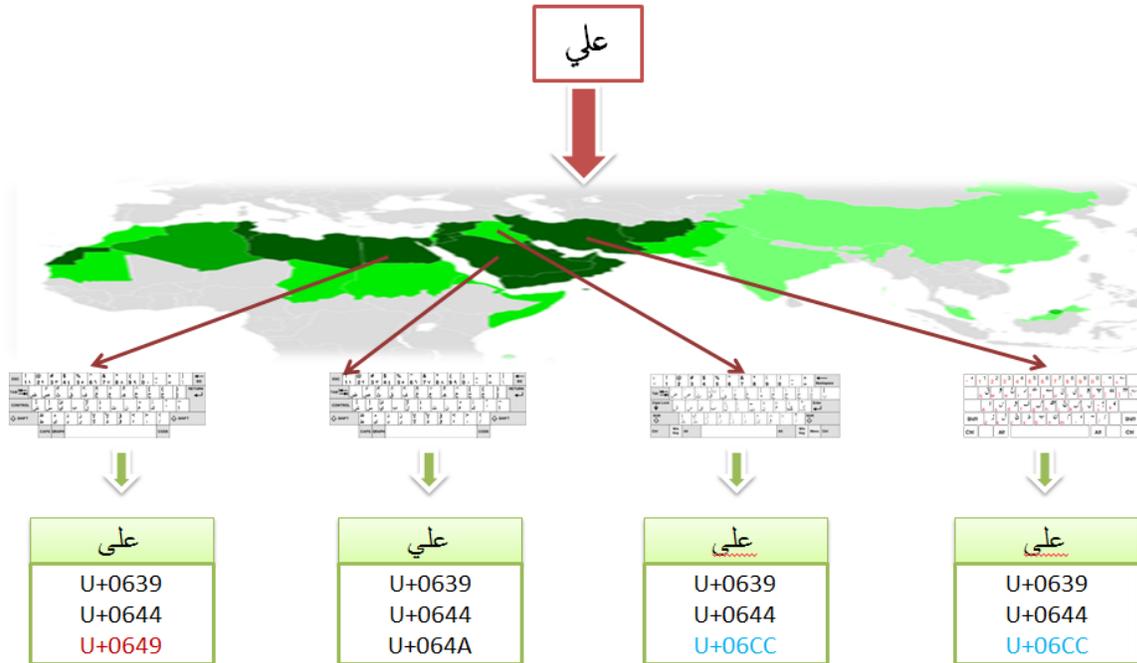


- Some Arabic characters have different shapes depending on their use. For example, in the Arabic Language:
  - There are 28 letters ... but a normal keyboard has about 40 different letter keys, e.g., there are 4 forms of Alef
  - Some letters are used interchangeably due to:
    - Different writing styles, e.g., the use of (ي ، ي) at the end of words.
    - The use of (ـ) instead of (ة) at the end of words as they have the same sound
- In normal writing (including newspapers, web site, social media, ...) people do not make a distinction between different forms of Alef (آ، إ، أ، ا). For example, the word "Internet" is written in most Arab countries in north Africa as (أنترنت) while it is written in other Arab countries as (إنترنت), however, end users often write it as (انترنت). Additionally, many words have two correct ways to write them. For example, (أدم and آدم) are widely used for the same name Adam.

Therefore, users (navigators or registrants) are expecting to reach a domain name regardless of how it was typed and there should not be any assumption regarding the user/applicant 1<sup>st</sup> choice. Hence the variant management system should provide the user of the ability to correct his/her 1<sup>st</sup> choice for the purpose of reachability and stability.



## Supporting and Managing Arabic Domain Names



One main objective of a registry is to reduce the huge size of allocate-able variants by intelligently identify and displaying only the desired variants. Therefore, the concept “variant filters” has been introduced to achieve this objective by filtering-out undesired variants and suggest some desired variants to the registrants.

Filters are based on linguistic study of Arabic words to find some rules that would help identifying desired variants. For example, N-grams models can be used to study the repetitive patters in words. An example of 2-gram for the word “ cars ” would be: “ c”, “ca”, “ar”, “rs”, and “s “. Filters can be combined with a ranking system to arrange allocate-able variants based on weight given by each rule. Rules should be confirmed with linguists and researchers.

SaudiNIC studied 2, 3 and 4-grams for more than 7 million non-repetitive words in the Arabic language using different sources such as books, newspapers, refereed academic journals, which were part of KACST's Arabic Corpus. A number of high-frequency patterns have been studied (e.g., آل\*, آل\*, ... etc.) and then built some rules/filters based on them.



SaudiNIC has developed more than 25 filters that tremendously and massively helped filtering out undesired variants and hence reduce the number of possible choices for allocate-able variants.

The following table depicts some filters (designed specifically for the Arabic language) along with their brief descriptions and some examples of filtered-out items.

Pattern	Description	Input	Filtered out
<b>Filter Name: AlefMadaEnd</b>			
ALEF WITH MADDA ABOVE at the end of a word: ā*	This filter identifies variants with the specified pattern and then removes them, i.e., an ALEF WITH MADDA ABOVE is seldom to come at the end of words.	خطا-ظماً	خطا-ظماً خطا-ظماً خطا-ظماً ...
<b>Filter Name: AlefHamzaDownEnd</b>			
ALEF WITH HAMZA BELOW at the end of a word: ǎ*	This filter identifies variants with the specified pattern and then removes them, i.e., an ALEF WITH HAMZA BELOW is seldom to come at the end of words.	خطا-ظماً	خطا-ظماً خطا-ظماً خطا-ظماً ...
<b>Filter Name: Al-Tarif</b>			
ALEF WITH HAMZA BELOW, or ALEF WITH MADDA ABOVE, or ALEF WITH HAMZA ABOVE followed by LAM (*أل, *آل, *إل) at the beginning of a word.	This filter identifies variants with the specified patterns and then removes them, i.e., most words that start with (ال) do not need other variants.	القرآن	أَلقرآن إَلقرآن أَلقرآن ...
<b>Filter Name: 2AlefwithChr</b>			
ALEF WITH HAMZA BELOW, or ALEF WITH MADDA ABOVE, or ALEF WITH HAMZA ABOVE followed by one character then followed by ALEF WITH HAMZA BELOW, or ALEF WITH MADDA ABOVE, or ALEF WITH HAMZA ABOVE (e.g., *أأ*)	This filter identifies variants with the specified patterns and then removes them.	رايات	رأيات رأيات رأيات رأيات رأيات رأيات ...
<b>Filter Name: AlefTaaEnd</b>			
ALEF WITH HAMZA BELOW, or ALEF WITH MADDA ABOVE, or ALEF WITH HAMZA ABOVE followed by The (*أت, *آت, *إت)	This filter identifies variants with the specified pattern and then removes them, i.e., these patterns are seldom to come at the end of words.	اتصالات	اتصالات اتصالات اتصالات ...

To test and see the behavior of the implemented filters uses the publicly available tools: [SaudiNIC's Variants Management System \(DEMO\)](#).



**Benefits of the Concept** (Variant Filters):

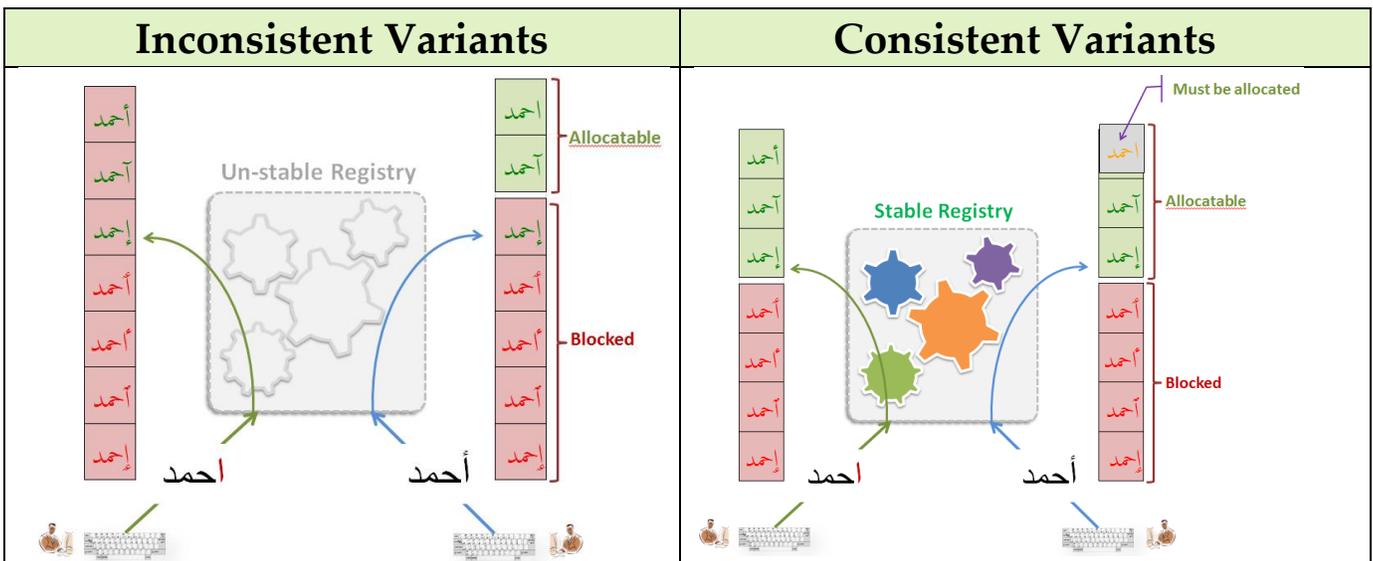
- Remarkably help filtering out undesired variants and hence reduce the number of possible choices for allocate-able variants.
- Help and simplify the choosing process of variants (Giving the user the best desired variants).
- Undesired variants still can be allocated through other ways based on the registry choice.

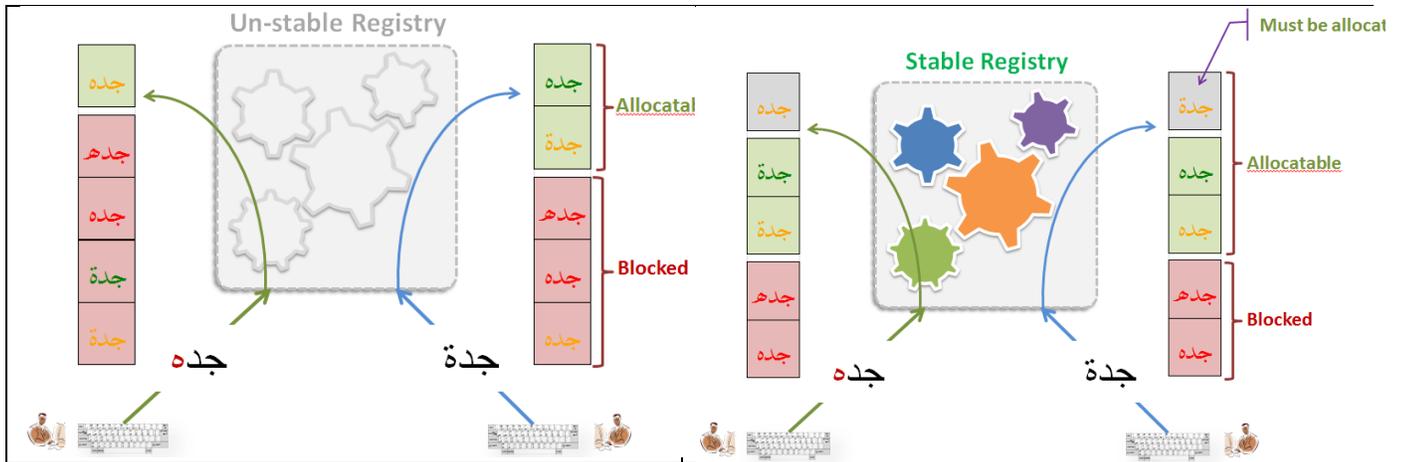
3.1.12. Consistent Variants

Consistency is very important concept in variants generation. Regardless of the selected applied-for label the list of generated allocate-able variants should be the same. Please note that the 1<sup>st</sup> choice (applied-for) label entered by a normal user, might not be the one will be used by the internet community. Therefore, a registry should provide the registrant the possibility to "correct" his/her choice if he/she was not successful with the first try.

For example, as stated before the word "Internet" might be written in 3 different forms (إنترنت، أنترنت، انترنت), then if someone registered "أنترنت", he/she should be able to enable "إنترنت" or "انترنت". Likewise, if someone registered "أدم" he/she should be able to enable "آدم" or "ادم".

The following chart shows behaviors of different registries depending on the applied-for labels entered by applicants. As it can be seen, the right side (stable registry) generates the same allocate-able variants regardless of the selected applied-for label. While, the "un-stable" registry (left-side) generates different allocate-able variants for different selected applied-for labels.





## Benefits of the Concept (Consistent Variants):

- Assist in having stable and secure registry solution.
- Meet user expectations
- Simplify the registration process (and hence make the registrant life easy)
- Enhance both user and universal acceptance

## 3.2. Practices

The following subsections are some general practices that are needed to be followed by registries to attain the necessary requirements to support and manage Arabic IDNs. They will include also some summaries of SaudiNIC's current practices.

### 3.2.1. Identifying Supported Language(s)

The registry need to choose which language(s) that will be supported under their TLD. It should start by supporting only ready and mature languages that have language and variant tables already defined (i.e., exercising the concept [Step-by-Step Language Support](#)). Other languages can be added at any time whenever they get ready (by having their language and variant tables being defined).

#### **SaudiNIC's Current Practice:**

The Arabic language is the main supported language for the IDN ccTLD (السعودية). that are managed by SaudiNIC. Hence, SaudiNIC started by supporting the Arabic language first and then gradually added other languages (currently 4 of them: Persian, Urdu, Pashto, and Malay). They have been selected based on the following criteria:

- There should be a language and variant tables available that are defined and published on the Internet by a recognized and qualified party from the corresponding community or a related ccTLD operator.
- There is a well-known input device (keyboard) that supports the code points listed in the language table.
- There are some online contents writing in the chosen language.



### 3.2.2. Acquiring Language and Variant Tables

For each supported language, the registry should obtain or coordinate with associated language communities (or language experts) to achieve the following:

- [A Language Table](#).
- [A Variant Table](#).

Language tables are very useful tools to registries. They can be used in the registration system interface to verify user input. They are also utilized to employ the powerful concept “[no language mixing](#)”. This concept will aid in identifying “[must-be-allocated](#)” variants for reachability purposes. Furthermore, this concept will also identify blocked variants due to language mixing that will tremendously contribute in reducing the number of unnecessary allocatable variants.

A variant table should be based on studying the similarities for the supported language code points [across the whole script](#) and identifying how users may type a domain name [using different input devices](#) from other languages. It should determine the [similarity relationship types \(Exact, Typo\) and actions](#) (Allocated, must-be-allocated, Blocked)

#### **SaudiNIC's Current Practice:**

*The Arabic language table developed by SaudiNIC was generated based on the [RFC 5564](#) (entitled: “Linguistic Guidelines for the Use of the Arabic Language in Internet Domains”) which was a community effort that passed through a number of stages:*

- *First step involved identifying a number of Arabic linguistic issues with respect to domain names*
- *These issues discussed among more than 60 members involved in the linguistic committee of the Arab Internet Names Consortium (AINC) until final recommendations were reached.*
- *A number of online questionnaires was done to survey users' opinions in these issues. More than 500 responses were received from participants. All had been analyzed and compared with the recommendations of the linguistic committee.*
- *These recommendations were discussed in face-to-face meetings with a number of Arabic linguists to get their guidance and assure the suitability of the recommendations.*
- *The process and recommendations were presented at the assembly meeting of the Saudi Arabic Language Association.*
- *These issues and recommendation were discussed and endorsed by the Arab team for domain names, which was formed under the auspices of the Arab League.*
- *This continuous and persistent effort had led to the publication of an RFC 5564: <https://tools.ietf.org/html/rfc5564>.*



When the language table was being constructed, the focus was on developing a simple character set table that includes ONLY needed letters and digits. It observed the following principles:

- It should be a group (community) effort and it was done through studying and discussing the linguistic issues extensively (more than 3 years) by a team who worked under the umbrella of Arab League.
- Follow the inclusion model recommended by the IDNA standard by adhering to LDH convention as many characters are disallowed, e.g., symbols , punctuations, diacritics (tashkeel), Koranic annotation signs, honorifics, etc.
- Arabic labels are not intended to write sentences or phrases.

The tables in ([Appendix 1](#) (Language tables) : [Arabic Language table](#)) constructed using an inclusion-based approach. Thus, characters that are not part of these tables are prohibited.

SaudiNIC has studied variants in three stages or levels:

1. Variants within the language:  
Variants have been identified between code points within the language after coordinating directly with linguist experts and observing the common use within our community. After identifying variant characters within the language, they have been reviewed and confirmed by linguist experts. Some additional variants haven added for protection and safeguard the namespace (such as the number five with Heh and the number 1 with Alef)
2. Variants with the rest code points of the Arabic script:  
Two methods (manual, image-comparison software) were used to identify variants by generating of all forms of character positions (start, end, middle, isolated) of the Arabic language code points and then comparing them (from Arab user point of view) with the rest of the forms and positions other Arabic script code points. The results from both methods have been compared and verified that the whole process was thorough and complete.
3. Variants for International reachability  
This was done by studying how would the user (who has registered a domain name, or any person who wanted to access the domain name) access the domain name by writing it on a keyboard belonging to another language.

See the tables in ([Appendix 2: Arabic language table in IANA new XML format](#)) .

### **Supplementary Notes:**

1. Warning concerning adding "forging" letters to the Arabic Alphabet:  
There have been some odd and nonstandard demand to add some foreign (alien) letters to the Arabic language, such as the following foreign letters: كُ



ج، ث، ..etc). Therefore, SaudiNIC's practice with respect to this issue is summarized in the following points:

- Language communities in the Arab world agreed on the official letters for the Arabic language and the Arab League States (LAS) consulted its member before publishing RFC5564.
- The Arab domain name team (under LAS) published RFC5564 in cooperation with ESCWA (United Nations).
- LAS coordinated with the "Egyptian Organization for Standards & Quality" (EOS) to issue a standard (Number: 6996/2009) that documented the code points for the Arabic Language which are identical to the code points listed in the RFC5564.
- These foreign letters (eg., ج، ث، ك) are not part of the official Arabic language, not taught in schools, and not used by official newspapers (even in the countries that are known for using them).
- The registry managers in the countries that are known for using these foreign letters have not included them as acceptable code points for registering domain names under their IDN ccTLD (e.g. الجزائر، تونس، مصر).
- Keyboard layouts for well-known operating systems that support the Arabic language (e.g. Microsoft, Apple, Linux, Android ..etc.) do not include these foreign letters. Thus, basic Arab users might face problems to type them even if they are known to him. Moreover, other Arab users who does not know these foreign letters will not be able to reach the domain/website and the registrant will face reachability problems.

Therefore, it is strongly recommended to limit the code points for the Arabic language to the code points that was defined in the RFC5564. If any country/community still believe that these foreign letters are needed then a new separate Language table should be created but should not be part of the Arabic Language table.

### 2. Warning concerning the use of hidden and special control characters:

There are some control characters, which are not known for the Arab Communities, however other communities (such as Persian and Urdu) may know and use them. The control characters are:

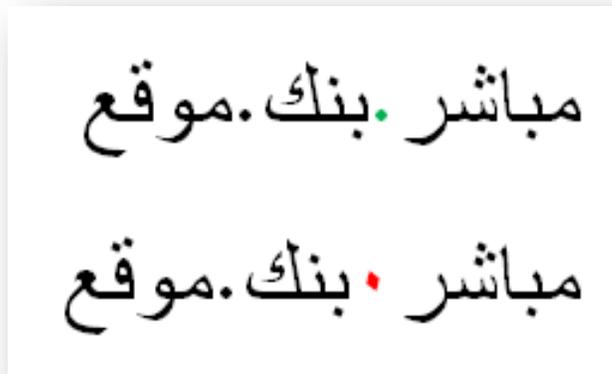
- Zero-Width-Joiner (ZWJ): When placed between two characters that would otherwise not be connected, a ZWJ causes them to be printed in their connected forms.



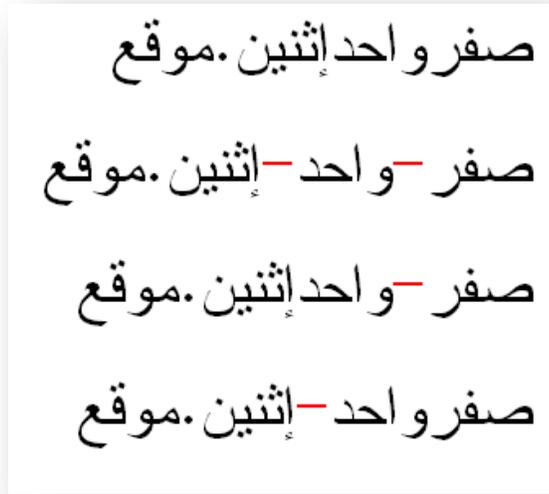
- *Zero-Width-Non-Joiner (ZWNJ): When placed between two characters that would otherwise be connected into a ligature, a ZWNJ causes them to be printed in their final and initial forms, respectively.*
- *In some cases those control character has no effect on the shape of the character it follows and hence it cause confusion with the same character sequence when ZWNJ is not used. SaudiNIC highly recommends not supporting these control characters due to the following reasons:*
  - *Allowing ZWNJ in domain names is a very serious problem and threat to our users.*
  - *ZWNJ causes some security, stability, usability, and reachability problems, and hence mistrust of IDNs.*
  - *At Script level, the ZWNJ is considered by UNICODE to be an invisible join control character and listed in the "Unicode Security Considerations" document, which warns that incorrect usage can expose programs or systems to possible security attacks. This is especially relevant for IDNs.*
  - *The ZWNJ in some cases is not visible to all users (e.g., U+0637, U+0638, U+069F, U+06BE, and U+06FF). A comprehensive analysis of Unicode Arabic Script Code Charts is needed to find any additional cases. This process should be repeated as the Unicode gets updated.*
  - *The ZWNJ concept and behavior are not known to many Arabic script users, who do not use it or know how to type it.*
  - *ZWNJ is not conveniently available on the keyboard, where typing it requires multiple simultaneous key-presses, which is complicated for users. ZWNJ is also inconsistently placed on keyboards across various operating systems. In addition, it is not available on many keyboards, making it difficult for people to use ZWNJ when, for example, they travel.*
  - *The users may not be able to type a domain name as they may think it is a <space> not ZWNJ, which may lead to reachability and usability problems, and therefore, mistrust of IDNs.*
  - *Based on the ZWNJ Contextual Rule (RFC 5892 Appendix A.1) for handling CONTEXTJ labels under the current IDNA2008, the Rule implementation does not totally resolve the non-visibility problem particularly in some cases as discussed above (e.g., U+0637, U+0638,U+069F, U+06BE, and U+06FF).*



- It is not one with the general category of {Ll, Lo, Lm, Mn}, as per the requirement defined by the gTLD Applicant Guidebook (v 2011-09-19, Module 2, page 2-13, Section 2.2.1.3.2, Part II, Item 2.1.3.).
  - Use of ZWNJ may cause additional bidirectional display issues.
  - Hyphen can be used instead of ZWNJ to break a string into ligatures
  - For more detail, see [Appendix 4: Control Characters: ZWJ and ZWNJ](#)
3. Notes concerning whether (و) is variant with (ؤ) or (ى) is variant with (ئ):  
After consulting linguist experts, SaudiNIC reached the conclusions that the code points [0624(ؤ) & 0648(و)] and [0626(ئ) & 064A(ي)] are not variants for the following reasons:
- From linguistic point of view, they are not the same and should not be considered as variants. As 0624(ؤ) and 0626(ئ) are considered to be a form of Hamza.
  - Both 0648(و) and 064A(ي) can occur at the beginning, middle and end of the word. However 0624(ؤ) and 0626(ئ) can only occur at the middle and end of a word (does not occur at the beginning).
  - In certain cases 0624(ؤ) and 0626(ئ) may be confused with each other (for example: شئون، شؤون، مسئول، مسؤول) but not with 0648(و) or 064A(ي) :
4. List of some other possible variants to be considered for security reasons:
- Arabic zero with dot:



- Hyphen (as a word separator)



### Other Supported Languages:

SaudiNIC has obtained language and variant tables for other supported languages (Persian, Urdu, Pashto, and Malay) from recognized and appreciated entities from the corresponding communities. These tables are shown in Appendix 1 (Language tables) .:

	Language	Source
1	Persian	See <a href="#">Appendix 1: Persian language table</a> Source: - <a href="http://www.nic.ir/Allowable_Characters_dot-iran">http://www.nic.ir/Allowable_Characters_dot-iran</a>
2	Urdu	See <a href="#">Appendix 1: Urdu language table</a> Source: - <a href="http://www.panl10n.net/english/Outputs%20Phase%202/RRCs/RRC-Technology/IDNFeedbackofPANL10nproject_v1.01.pdf">http://www.panl10n.net/english/Outputs%20Phase%202/RRCs/RRC-Technology/IDNFeedbackofPANL10nproject_v1.01.pdf</a> - <a href="http://www.cle.org.pk/IDN/IDN2010/minutesofmeeting.pdf">http://www.cle.org.pk/IDN/IDN2010/minutesofmeeting.pdf</a> - <a href="https://registry.in/system/files/INTERNATIONALIZED_DOMAIN_NAMES-URDU.PDF">https://registry.in/system/files/INTERNATIONALIZED_DOMAIN_NAMES-URDU.PDF</a>
3	Pashto	See <a href="#">Appendix 1: Pashto language table</a> Source: - <a href="http://www.panl10n.net/english/Outputs%20Phase%202/RRCs/RRC-Technology/IDNFeedbackofPANL10nproject_v1.01.pdf">http://www.panl10n.net/english/Outputs%20Phase%202/RRCs/RRC-Technology/IDNFeedbackofPANL10nproject_v1.01.pdf</a> - <a href="http://www.khpalapashtu.com/sitee/pashtula/pasalph.htm">http://www.khpalapashtu.com/sitee/pashtula/pasalph.htm</a>
4	Malay	See <a href="#">Appendix 1: Malay (Jawi) language table</a> Source: - <a href="http://www.iana.org/domains/idn-tables/tables/my_ms-my_1.0.pdf">http://www.iana.org/domains/idn-tables/tables/my_ms-my_1.0.pdf</a> - <a href="http://en.wikipedia.org/wiki/Jawi_alphabet">http://en.wikipedia.org/wiki/Jawi_alphabet</a>



### 3.2.3. *Resolving Variant Conflicts and Finalizing Variant Tables*

The registry then should resolve any conflicts in variants or variant types that occurred from integrating “language and variant tables”. For example, if a language community decided two letters are variants while another language community believe that those two letters are not variants. The registry should put some rules to solve such cases:

- conflicts in variant types: Language 1: A = B , Language 2: A <> B
- conflicts in variant actions: Language 1: C = D (Blocked) , Language 2: C = D (Allocated)

Eventually, the registry should by now finalize language and variant tables for all the supported languages.

#### **SaudiNIC's Current Practice:**

The following manners have been followed by SaudiNIC in case of conflicts:

- There should be no side effects on registrants/users.
- Protecting the namespace and the registry
- In case of a conflict between two language communities on the existence of similarity between two code points. One community claims there is a similarity between the two code points and they should be consider variants, while the other community doesn't. Then, for the propose of safeguard the namespace the two code points should be considered as variants.
- In case the conflict is in the action. Where one community suggests allocation while the other suggests blocking. Then, the variants should be allocateable so that no damage on the language community.

### 3.2.4. *Implement Variant Keys*

By now (i.e., after completing the above practices) the registry has managed to achieve all the basic requirements which will aid it to secure its IDN TLD name space. For example, the following components will aid the registry to:

- Supported language tables
  - List of code points for each language will assist in controlling input string from registrants.
  - Will be used to stop mixing characters from different languages, i.e., it will dramatically reduce the number of allocateable variants as variants due to language mixing will be blocked.
- Variant Table
  - Variants, Variants Types/Actions (e.g. Allocate-able, Blocked)
  - Will be used to generate allocate-able variants



Additionally, the registry should implement and use a mechanism to secure variants from being registered by others. It dose that by grouping variants under [one key](#).

### **SaudiNIC's Current Practice:**

*SaudiNIC has developed a "Master Key algorithm" which simply integrates all supported language tables together and resolve any conflicts between them then build key components that will group variants under one key. It also makes adding a new language table in the future an easy task which will minimize or eliminate the need to rebuild the keys for already registered domain names. For more technical information see:*

[http://arabic-domains.org/docs/Master\\_Key\\_Algorithm.pdf](http://arabic-domains.org/docs/Master_Key_Algorithm.pdf)

### **3.2.5. Provide clever tools to manage variants (VMS)**

The registry should provide a variants management system that is simple and clever to help Registrants with the following tasks:

- Register a domain name in their language
- List allocate-able and/or desired variants
- Enable and disable allocate-able variants
  - It is recommended to build the necessary filters to simplify displaying desired variants to the registrants. Filters will be used to suggest desired variants for the purpose to reduce the huge size of allocate-able variants by intelligently displaying only the desired variants.

### **SaudiNIC's Current Practice:**

*The following tool gives you the opportunity to test our solution. It generates all possible variants (allocate-able and blocked) based on some of the supported languages (Arabic, Persian, Urdu, Malay and Pashto). It also can filter out the allocate-able variants into desired and undesired variants and identify the must be allocated variants:*

[http://arabic-domains.org/adn\\_tools/mk/index.php](http://arabic-domains.org/adn_tools/mk/index.php)

*The technical description of the Master Key algorithm that are used in our proposal can be found in the following links:*

[http://arabic-domains.org/docs/Master\\_Key\\_Algorithm.pdf](http://arabic-domains.org/docs/Master_Key_Algorithm.pdf)

<http://nic.sa/en/view/doc64>

*Please note, the above "public" tools and documentations were developed and written some time ago (2007-2010), therefore some common terminologies are not yet developed at that time so we use our own terminologies that you may have difficult to grasp from the first reading. SaudiNIC is continuously modifying*

## Supporting and Managing Arabic Domain Names



these tools and incorporating within them all the new developed concepts and ideas but they are kept for internal usage until it reaches acceptable and stable stage when they become ready to publish for the public. Hence, SaudiNIC in the process of customizing and polishing these tools to put them for public use as well as updating the related documents to use the new terminologies and to support the proposed format outlined in the "Representing registration policy for IDNs using XML" that will be used in IANA Repository.

SaudiNIC developed a set of filters to divide the allocate-able variants into two sets: desired variants (limited list and optimized for the user) and undesired variants (huge list and mostly the user will not ask for them).

The following is a snapshot of our Variant Management System (old version):

معلومات عن النطاق  
اسم النطاق: أمثلة-شركة.السعودية

مثل للتشبيهاً  
اسم النطاق باستخدام أمثلة شركة.السعودية  
المسافات (بدون ارجح اسم النطاق مستخدماً "الفراغ" للفصل بين الكلمات (من دون شرطيات) شرطات  
المتال الأول: أمثلة-شركة.السعودية  
المتال الثاني: أمثلة-شركة.السعودية  
المتال الثالث: أمثلة-شركة.السعودية  
المتال الرابع: أمثلة-شركة.السعودية

قائمة التشبيهاً الجديدة  
 حذف جميع التشبيهاً

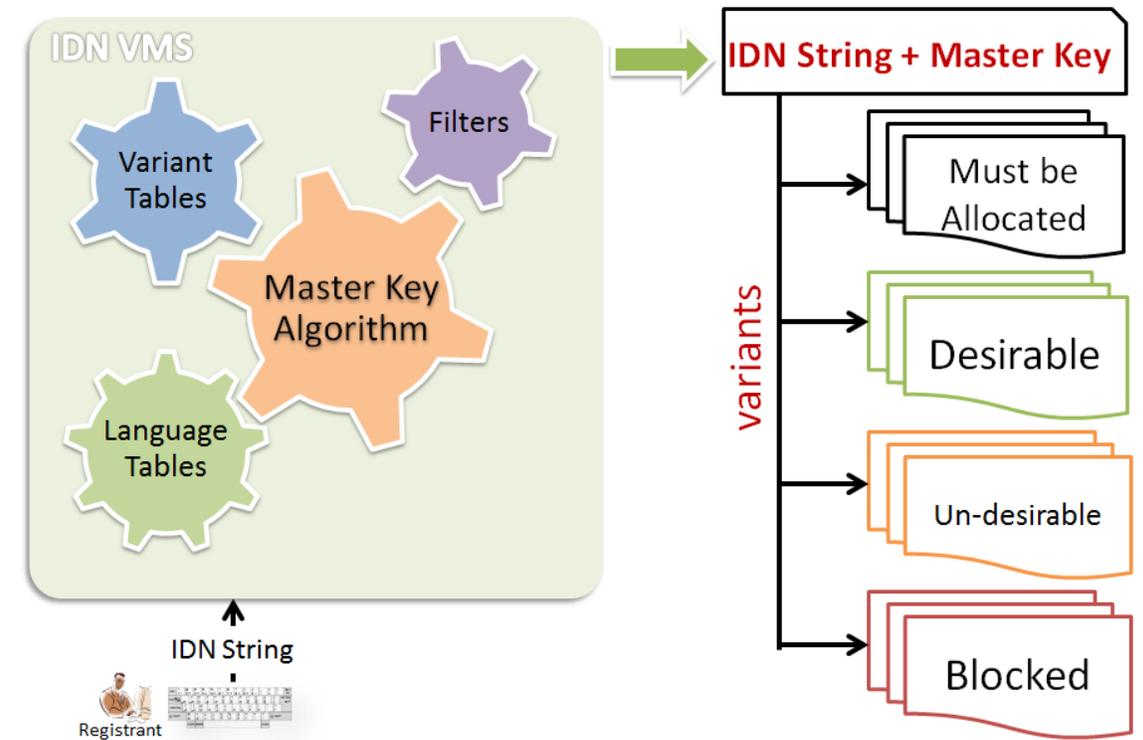
الترتيب الأول: السعودية  
الترتيب الثاني: السعودية  
الترتيب الثالث: السعودية

Domain Name  
Domain name: أمثلة-شركة.السعودية

Example For Variants  
Domain Name with spaces: أمثلة شركة.السعودية Show Examples  
Enter the domain name using a "space" character between words (no dashes).  
Example one: أمثلة-شركة.السعودية  
Example two: أمثلة-شركة.السعودية  
Example three: أمثلة-شركة.السعودية  
Example four: أمثلة-شركة.السعودية

New Variants  
Delete all variants:

First variant: .السعودية  
Second variant: .السعودية  
Third variant: .السعودية



## 4. Other Considerations

This section highlights some issues in different field that have to be considered to support and manage Arabic domain names.

### 4.1. Regulation Issues

There has to be a set of good regulations that controls the registration and usage of IDN such as defining what are the accepted code points (Language Tables) and what are the variants and what can be allocated and what are blocked ...etc. (See [Appendix 3: Guideline Rules for writing Arabic Labels under \(السعودية\)](#)).

Here is an example list (not comprehensive) of important issues that should be included in the registration policy:

- Variants are grouped together and associated to only one registrant, i.e., they cannot be separated. Thus, it's not possible to have a variant with one registrant and another variant (belongs to the same group) to another registrant.
- The registrant is responsible (and liable) for any objection or dispute regarding his domain name and its' variants.
- The registrant should not enable/activate any variants that infringe the right of a third party.
- Variants can only be used by the same registrant.



## 4.2. Reserved List Issues

When creating and managing reserved list of IDN word and phrases, their variants should be also part of the reserved list. That is when a label is not available for registration because it is part of the reserved list, all variants of this label must be blocked as well.

## 4.3. Technical issues

There are many technical considerations in the registry systems that should be considered in order to support Arabic domain name.

The following items are some examples of those considerations:

- The registry system should enable the registrant to register Arabic domain names and verify the inputs agents the supported language tables, also it should and accepts Arabic name servers and Arabic email address for the registrant contacts.
- The registry systems should verify the domain name is available for registration by checking the registered domain names and the all possible variants of the registered domain names using the master key algorithm or any similar mechanism.
- The Whois service should list the Arabic domain name as both U-Label and A-Label (punycode representation) and the activated variants.
- The zone builder should make sure to publish the A-label and not the U-label for the domain names and the name server.
- The registry system should provide necessary EPP extension and user interface that support IDN and support listing, enabling and disabling variants.

New IDN Registries must inform their clients how to activate IDN in some browsers , also the registries must contact major browsers venders to make sure the support for their IDN is available, for example any new IDN registry must add it's ( Public suffix ) to Mozilla records (<https://publicsuffix.org> ) in order to ensure full support from the applications .

Arabic script IDN registries should coordinate between each other continuously along with exchanging the gained experiences about IDNs in order to meet the user expectations and trust along with keeping their TLDs name space secure and safe.

These are some examples of situations that may adversely affect the user trust in Arabic domain names:

- Different language tables between registries for the same language community (this include the supported code points and their variants).
- Different mechanisms for enabling and disabling variant (this include the number of variants that can be enables and the action on each one of them).
- Different terms and terminologies used to describe the variants and their actions.



## Appendix 1 (Language tables) :

## I. Arabic Language table

code	Character/symbol	Unicode name
0621	ء	ARABIC LETTER HAMZA
0622	أ	ARABIC LETTER ALEF WITH MADDA ABOVE
0623	آ	ARABIC LETTER ALEF WITH HAMZA ABOVE
0624	ؤ	ARABIC LETTER WAW WITH HAMZA ABOVE
0625	إ	ARABIC LETTER ALEF WITH HAMZA BELOW
0626	ئ	ARABIC LETTER YEH WITH HAMZA ABOVE
0627	ا	ARABIC LETTER ALEF
0628	ب	ARABIC LETTER BEH
0629	ة	ARABIC LETTER TEH MARBUTA
062A	ت	ARABIC LETTER TEH
062B	ث	ARABIC LETTER THEH
062C	ج	ARABIC LETTER JEEM
062D	ح	ARABIC LETTER HAH
062E	خ	ARABIC LETTER KHAH
062F	د	ARABIC LETTER DAL
0630	ذ	ARABIC LETTER THAL
0631	ر	ARABIC LETTER REH
0632	ز	ARABIC LETTER ZAIN
0633	س	ARABIC LETTER SEEN
0634	ش	ARABIC LETTER SHEEN
0635	ص	ARABIC LETTER SAD
0636	ض	ARABIC LETTER DAD
0637	ط	ARABIC LETTER TAH
0638	ظ	ARABIC LETTER ZAH
0639	ع	ARABIC LETTER AIN
063A	غ	ARABIC LETTER GHAIN
0641	ف	ARABIC LETTER FEH
0642	ق	ARABIC LETTER QAF
0643	ك	ARABIC LETTER KAF
0644	ل	ARABIC LETTER LAM
0645	م	ARABIC LETTER MEEM
0646	ن	ARABIC LETTER NOON
0647	هـ	ARABIC LETTER HEH
0648	و	ARABIC LETTER WAW
0649	ى	ARABIC LETTER ALEF MAKSURA
064A	ي	ARABIC LETTER YEH
0660	0	ARABIC-INDIC DIGIT ZERO
0661	1	ARABIC-INDIC DIGIT ONE
0662	2	ARABIC-INDIC DIGIT TWO
0663	3	ARABIC-INDIC DIGIT THREE
0664	4	ARABIC-INDIC DIGIT FOUR
0665	5	ARABIC-INDIC DIGIT FIVE
0666	6	ARABIC-INDIC DIGIT SIX
0667	7	ARABIC-INDIC DIGIT SEVEN
0668	8	ARABIC-INDIC DIGIT EIGHT
0669	9	ARABIC-INDIC DIGIT NINE
0030	0	DIGIT ZERO
0031	1	DIGIT ONE
0032	2	DIGIT TWO
0033	3	DIGIT THREE
0034	4	DIGIT FOUR
0035	5	DIGIT FIVE
0036	6	DIGIT SIX
0037	7	DIGIT SEVEN
0038	8	DIGIT EIGHT
0039	9	DIGIT NINE
002D	-	HYPHEN-MINUS

Source: <https://tools.ietf.org/rfc/rfc5564.txt>



## II. Urdu language table

code	Character/symbol	Unicode name
0621	ء	ARABIC LETTER HAMZA
0622	آ	ARABIC LETTER ALEF WITH MADDA ABOVE
0623	أ	ARABIC LETTER ALEF WITH HAMZA ABOVE
0624	ؤ	ARABIC LETTER WAW WITH HAMZA ABOVE
0627	ا	ARABIC LETTER ALEF
0628	ب	ARABIC LETTER BEH
062A	ت	ARABIC LETTER TEH
062B	ٹ	ARABIC LETTER THEH
062C	ج	ARABIC LETTER JEEM
062D	ح	ARABIC LETTER HAH
062E	خ	ARABIC LETTER KHAH
062F	د	ARABIC LETTER DAL
0630	ذ	ARABIC LETTER THAL
0631	ر	ARABIC LETTER REH
0632	ز	ARABIC LETTER ZAIN
0633	س	ARABIC LETTER SEEN
0634	ش	ARABIC LETTER SHEEN
0635	ص	ARABIC LETTER SAD
0636	ض	ARABIC LETTER DAD
0637	ط	ARABIC LETTER TAH
0638	ظ	ARABIC LETTER ZAH
0639	ع	ARABIC LETTER AIN
063A	غ	ARABIC LETTER GHAIN
0641	ف	ARABIC LETTER FEH
0642	ق	ARABIC LETTER QAF
0644	ل	ARABIC LETTER LAM
0645	م	ARABIC LETTER MEEM
0646	ن	ARABIC LETTER NOON
0648	و	ARABIC LETTER WAW
0679	ٹا	ARABIC LETTER TTEH
067E	پ	ARABIC LETTER PEH
0686	ھ	ARABIC LETTER TCHEH
0688	ڈ	ARABIC LETTER DDAL
0691	ڑ	ARABIC LETTER RREH
0698	ژ	ARABIC LETTER JEH
06A9	ک	ARABIC LETTER KEHEH
06AF	گ	ARABIC LETTER GAF
06BA	ں	ARABIC LETTER NOON GHUNNA
06BE	ھ	ARABIC LETTER HEH DOACHASHMEE
06C1	ہ	ARABIC LETTER HEH GOAL
06C2	وہ	ARABIC LETTER HEH GOAL WITH HAMZA ABOVE
06C3	ہ	ARABIC LETTER THE MARBUTA GOAL
06CC	ی	ARABIC LETTER PERSIAN YEH
06D2	ے	ARABIC LETTER YEH BARREE
06D3	ئے	ARABIC LETTER YEH BARREE WITH HAMZA ABOVE
06F0	۰	EXTENDED ARABIC-INDIC DIGIT ZERO
06F1	۱	EXTENDED ARABIC-INDIC DIGIT ONE
06F2	۲	EXTENDED ARABIC-INDIC DIGIT TWO
06F3	۳	EXTENDED ARABIC-INDIC DIGIT THREE
06F4	۴	EXTENDED ARABIC-INDIC DIGIT FOUR
06F5	۵	EXTENDED ARABIC-INDIC DIGIT FIVE
06F6	۶	EXTENDED ARABIC-INDIC DIGIT SIX
06F7	۷	EXTENDED ARABIC-INDIC DIGIT SEVEN
06F8	۸	EXTENDED ARABIC-INDIC DIGIT EIGHT
06F9	۹	EXTENDED ARABIC-INDIC DIGIT NINE
0030	0	DIGIT ZERO
0031	1	DIGIT ONE
0032	2	DIGIT TWO
0033	3	DIGIT THREE
0034	4	DIGIT FOUR
0035	5	DIGIT FIVE
0036	6	DIGIT SIX
0037	7	DIGIT SEVEN
0038	8	DIGIT EIGHT
0039	9	DIGIT NINE
002D	-	HYPHEN-MINUS

Source: [http://www.panl10n.net/english/Outputs%20Phase%202/RRCs/RRCTechnology/IDNFeedbackofPANL10nproject\\_v1.01.pdf](http://www.panl10n.net/english/Outputs%20Phase%202/RRCs/RRCTechnology/IDNFeedbackofPANL10nproject_v1.01.pdf)



## III. Persian language table

code	Character/symbol	Unicode name
0621	ء	ARABIC LETTER HAMZA
0622	آ	ARABIC LETTER ALEF WITH MADDA ABOVE
0623	أ	ARABIC LETTER ALEF WITH HAMZA ABOVE
0624	ؤ	ARABIC LETTER WAW WITH HAMZA ABOVE
0626	ئ	ARABIC LETTER YEH WITH HAMZA ABOVE
0627	ا	ARABIC LETTER ALEF
0628	ب	ARABIC LETTER BEH
0629	ة	ARABIC LETTER TEH MARBUTA
062A	ت	ARABIC LETTER TEH
062B	ث	ARABIC LETTER THEH
062C	ج	ARABIC LETTER JEEM
062D	ح	ARABIC LETTER HAH
062E	خ	ARABIC LETTER KHAH
062F	د	ARABIC LETTER DAL
0630	ذ	ARABIC LETTER THAL
0631	ر	ARABIC LETTER REH
0632	ز	ARABIC LETTER ZAIN
0633	س	ARABIC LETTER SEEN
0634	ش	ARABIC LETTER SHEEN
0635	ص	ARABIC LETTER SAD
0636	ض	ARABIC LETTER DAD
0637	ط	ARABIC LETTER TAH
0638	ظ	ARABIC LETTER ZAH
0639	ع	ARABIC LETTER AIN
063A	غ	ARABIC LETTER GHAIN
0641	ف	ARABIC LETTER FEH
0642	ق	ARABIC LETTER QAF
0644	ل	ARABIC LETTER LAM
0645	م	ARABIC LETTER MEEM
0646	ن	ARABIC LETTER NOON
0647	ه	ARABIC LETTER HEH
0648	و	ARABIC LETTER WAW
067E	پ	ARABIC LETTER PEH
0686	چ	ARABIC LETTER TCHEH
0698	ژ	ARABIC LETTER JEH
06A9	ک	ARABIC LETTER KEHEH
06AF	گ	ARABIC LETTER GAF
06CC	ی	ARABIC LETTER PERSIAN YEH
06F0	۰	EXTENDED ARABIC-INDIC DIGIT ZERO
06F1	۱	EXTENDED ARABIC-INDIC DIGIT ONE
06F2	۲	EXTENDED ARABIC-INDIC DIGIT TWO
06F3	۳	EXTENDED ARABIC-INDIC DIGIT THREE
06F4	۴	EXTENDED ARABIC-INDIC DIGIT FOUR
06F5	۵	EXTENDED ARABIC-INDIC DIGIT FIVE
06F6	۶	EXTENDED ARABIC-INDIC DIGIT SIX
06F7	۷	EXTENDED ARABIC-INDIC DIGIT SEVEN
06F8	۸	EXTENDED ARABIC-INDIC DIGIT EIGHT
06F9	۹	EXTENDED ARABIC-INDIC DIGIT NINE
002D	-	HYPHEN-MINUS

## Supporting and Managing Arabic Domain Names



Source: [http://www.nic.ir/Allowable\\_Characters\\_dot-iran](http://www.nic.ir/Allowable_Characters_dot-iran)

### IV. Malay (Jawi) language table

code	Character/symbol	Unicode name
0621	ء	ARABIC LETTER HAMZA
0623	أ	ARABIC LETTER ALEF WITH HAMZA ABOVE
0625	إ	ARABIC LETTER ALEF WITH HAMZA BELOW
0626	آ	ARABIC LETTER YEH WITH HAMZA ABOVE
0627	ا	ARABIC LETTER ALEF
0628	ب	ARABIC LETTER BEH
0629	ة	ARABIC LETTER TEH MARBUTA
062A	ت	ARABIC LETTER TEH
062B	ث	ARABIC LETTER THEH
062C	ج	ARABIC LETTER JEEM
062D	ح	ARABIC LETTER HAH
062E	خ	ARABIC LETTER KHAH
062F	د	ARABIC LETTER DAL
0630	ذ	ARABIC LETTER THAL
0631	ر	ARABIC LETTER REH
0632	ز	ARABIC LETTER ZAIN
0633	س	ARABIC LETTER SEEN
0634	ش	ARABIC LETTER SHEEN
0635	ص	ARABIC LETTER SAD
0636	ض	ARABIC LETTER DAD
0637	ط	ARABIC LETTER TAH
0638	ظ	ARABIC LETTER ZAH
0639	ع	ARABIC LETTER AIN
063A	غ	ARABIC LETTER GHAIN
0641	ف	ARABIC LETTER FEH
0642	ق	ARABIC LETTER QAF
0644	ل	ARABIC LETTER LAM
0645	م	ARABIC LETTER MEEM
0646	ن	ARABIC LETTER NOON
0647	هـ	ARABIC LETTER HEH
0648	و	ARABIC LETTER WAW
0649	ى	ARABIC LETTER ALEF MAKSURA
064A	ي	ARABIC LETTER YEH
0686	چ	ARABIC LETTER TCHEH
06A0	ع	ARABIC LETTER AIN WITH THREE DOTS ABOVE
06A4	ف	ARABIC LETTER VEH
06A9	ك	ARABIC LETTER KEHEH
06F2	٢	EXTENDED ARABIC-INDIC DIGIT TWO
06BD	ن	ARABIC LETTER NOON WITH THREE DOTS ABOVE
06CF	و	ARABIC LETTER WAW WITH DOTS ABOVE
0762	ك	ARABIC LETTER KEHEH WITH DOT ABOVE
0030	0	DIGIT ZERO
0031	1	DIGIT ONE
0032	2	DIGIT TWO
0033	3	DIGIT THREE
0034	4	DIGIT FOUR
0035	5	DIGIT FIVE
0036	6	DIGIT SIX
0037	7	DIGIT SEVEN
0038	8	DIGIT EIGHT
0039	9	DIGIT NINE
002D	-	HYPHEN-MINUS

Source: [http://www.iana.org/domains/idn-tables/tables/my\\_ms-my\\_1.0.pdf](http://www.iana.org/domains/idn-tables/tables/my_ms-my_1.0.pdf)



## V. Pashto language table

code	Character/symbol	Unicode name
0621	ء	ARABIC LETTER HAMZA
0622	آ	ARABIC LETTER ALEF WITH MADDA ABOVE
0623	أ	ARABIC LETTER ALEF WITH HAMZA ABOVE
0624	ؤ	ARABIC LETTER WAW WITH HAMZA ABOVE
0626	ئ	ARABIC LETTER YEH WITH HAMZA ABOVE
0627	ا	ARABIC LETTER ALEF
0628	ب	ARABIC LETTER BEH
0629	ة	ARABIC LETTER TEH MARBUTA
062A	ت	ARABIC LETTER TEH
062B	ث	ARABIC LETTER THEH
062C	ج	ARABIC LETTER JEEM
062D	ح	ARABIC LETTER HAH
062E	خ	ARABIC LETTER KHAH
062F	د	ARABIC LETTER DAL
0630	ذ	ARABIC LETTER THAL
0631	ر	ARABIC LETTER REH
0632	ز	ARABIC LETTER ZAIN
0633	س	ARABIC LETTER SEEN
0634	ش	ARABIC LETTER SHEEN
0635	ص	ARABIC LETTER SAD
0636	ض	ARABIC LETTER DAD
0637	ط	ARABIC LETTER TAH
0638	ظ	ARABIC LETTER ZAH
0639	ع	ARABIC LETTER AIN
063A	غ	ARABIC LETTER GHAIN
0641	ف	ARABIC LETTER FEH
0642	ق	ARABIC LETTER QAF
0644	ل	ARABIC LETTER LAM
0645	م	ARABIC LETTER MEEM
0646	ن	ARABIC LETTER NOON
0648	و	ARABIC LETTER WAW
0660	0	ARABIC-INDIC DIGIT ZERO
0661	1	ARABIC-INDIC DIGIT ONE
0662	2	ARABIC-INDIC DIGIT TWO
0663	3	ARABIC-INDIC DIGIT THREE
0664	4	ARABIC-INDIC DIGIT FOUR
0665	5	ARABIC-INDIC DIGIT FIVE
0666	6	ARABIC-INDIC DIGIT SIX
0667	7	ARABIC-INDIC DIGIT SEVEN
0668	8	ARABIC-INDIC DIGIT EIGHT
0669	9	ARABIC-INDIC DIGIT NINE
067C	ٲ	ARABIC LETTER TEH WITH RING
067E	پ	ARABIC LETTER PEH
0685	ځ	ARABIC LETTER HAH WITH THREE DOTS ABOVE
0686	چ	ARABIC LETTER TCHEH
0689	ډ	ARABIC LETTER DAL WITH RING
0693	ړ	ARABIC LETTER REH WITH RING
0696	ږ	ARABIC LETTER REH WITH DOT BELOW AND DOT ABO
0698	ژ	ARABIC LETTER JEH
069A	ښ	ARABIC LETTER SEEN WITH DOT BELOW AND DOT ABOVE
06A9	ک	ARABIC LETTER KEHEH
06AF	گ	ARABIC LETTER GAF
06BC	ن	ARABIC LETTER NOON WITH RING
06C0	ه	ARABIC LETTER HEH WITH YEH ABOVE
06CD	ی	ARABIC LETTER YEH WITH TAIL
06D0	ی	ARABIC LETTER E
06D5	ه	ARABIC LETTER AE

Source: [http://www.pan10n.net/english/Outputs%20Phase%200/RRCs/RRCs-Technology/IDNFeedbackofPAN10nproject\\_v1.01.pdf](http://www.pan10n.net/english/Outputs%20Phase%200/RRCs/RRCs-Technology/IDNFeedbackofPAN10nproject_v1.01.pdf)



## Appendix 2: Arabic language table in IANA new XML format

```

<?xml version="1.0"?>
<lgr xmlns="urn:ietf:params:xml:ns:lgr-1.0">
  <meta>
    <version>2</version>
    <date>2016-06-05</date>
    <language>ar</language>
    <scope type="domain">xn--mgberp4a5d4ar</scope>
    <description type="text/plain">
      This document provides the IDN (Internationalized Domain Names) Language Table and
      guideline rules to be used for writing and registering Arabic Domain names under xn--mgberp4a5d4ar (السعودية) IDN
      ccTLD. These are based on the recommendations outlined in the RFC 5564: "Linguistic Guidelines for the Use of the
      Arabic Language in Internet Domains", that can be found in the following URL: http://tools.ietf.org/html/rfc5564
      Authors:
          Dr.Abdulaziz Al-Zoman (azoman[at]citc.gov.sa) , Saudi Network
          Information Center
          Raed Al-Fayez (rfayez[at]citc.gov.sa) , Saudi Network
          Information Center
    </description>
    <validity-start>2014-12-07</validity-start>
    <validity-end>2020-12-07</validity-end>
    <unicode-version>6.3.0</unicode-version>
    <references>
      <reference id="0">The Unicode Standard, Version 6.3.0</reference>
      <reference id="1">RFC 5564: "Linguistic Guidelines for the Use of the Arabic Language in
      Internet Domains", http://tools.ietf.org/html/rfc5564</reference>
      <reference id="2">Guideline Rules for writing Arabic IDNs under the IDN ccTLD (السعودية),
      http://nic.net.sa/docs/Guidelines\_for\_writing\_Arabic\_IDNs\_under\_the\_IDN\_ccTLD\_V1.2-en.pdf </reference>
    </references>
  </meta>
  <data>
    <char cp="0621" ref="0 1 2" />
    <char cp="0622" ref="0 1 2">
      <var cp="0623" type="allocate" when="arabic-final-right-join" comment="Language variant"/>
      <var cp="0623" type="allocate" when="arabic-isolated-right-join" comment="Language
      variant"/>
      <var cp="0625" type="allocate" when="arabic-final-right-join" comment="Language variant"/>
      <var cp="0625" type="allocate" when="arabic-isolated-right-join" comment="Language
      variant"/>
      <var cp="0627" type="activate" when="arabic-final-right-join" comment="Language variant"/>
      <var cp="0627" type="activate" when="arabic-isolated-right-join" comment="Language
      variant"/>
      <var cp="0671" type="block" when="arabic-final-right-join" comment="Typo variant"/>
      <var cp="0671" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
      <var cp="0672" type="block" when="arabic-final-right-join" comment="Typo variant"/>
      <var cp="0672" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
      <var cp="0675" type="block" when="arabic-final-right-join" comment="Typo variant"/>
      <var cp="0675" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
  </data>
</lgr>

```

## Supporting and Managing Arabic Domain Names



```
</char>

<char cp="0623" ref="0 1 2">
  <var cp="0622" type="allocate" when="arabic-final-right-join" comment="Language variant"/>
  <var cp="0622" type="allocate" when="arabic-isolated-right-join" comment="Language
variant"/>
  <var cp="0625" type="allocate" when="arabic-final-right-join" comment="Language variant"/>
  <var cp="0625" type="allocate" when="arabic-isolated-right-join" comment="Language
variant"/>
  <var cp="0627" type="activate" when="arabic-final-right-join" comment="Language variant"/>
  <var cp="0627" type="activate" when="arabic-isolated-right-join" comment="Language
variant"/>
  <var cp="0671" type="block" when="arabic-final-right-join" comment="Typo variant"/>
  <var cp="0671" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
  <var cp="0672" type="block" when="arabic-final-right-join" comment="Typo variant"/>
  <var cp="0672" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
  <var cp="0675" type="block" when="arabic-final-right-join" comment="Typo variant"/>
  <var cp="0675" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
</char>

<char cp="0624" ref="0 1 2">
  <var cp="0676" type="block" comment="Typo variant"/>
</char>

<char cp="0625" ref="0 1 2">
  <var cp="0622" type="allocate" when="arabic-final-right-join" comment="Language variant"/>
  <var cp="0622" type="allocate" when="arabic-isolated-right-join" comment="Language
variant"/>
  <var cp="0623" type="allocate" when="arabic-final-right-join" comment="Language variant"/>
  <var cp="0623" type="allocate" when="arabic-isolated-right-join" comment="Language
variant"/>
  <var cp="0627" type="activate" when="arabic-final-right-join" comment="Language variant"/>
  <var cp="0627" type="activate" when="arabic-isolated-right-join" comment="Language
variant"/>
  <var cp="0673" type="block" when="arabic-final-right-join" comment="Typo variant"/>
  <var cp="0673" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
</char>

<char cp="0626" ref="0 1 2">
  <var cp="0678" type="block" comment="Typo variant"/>
  <var cp="06D3" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
  <var cp="06D3" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
</char>

<char cp="0627" ref="0 1 2">
  <var cp="0622" type="allocate" when="arabic-final-right-join" comment="Language variant"/>
  <var cp="0622" type="allocate" when="arabic-isolated-right-join" comment="Language
variant"/>
  <var cp="0623" type="allocate" when="arabic-final-right-join" comment="Language variant"/>
  <var cp="0623" type="allocate" when="arabic-isolated-right-join" comment="Language
variant"/>
  <var cp="0625" type="allocate" when="arabic-final-right-join" comment="Language variant"/>
  <var cp="0625" type="allocate" when="arabic-isolated-right-join" comment="Language
variant"/>
</char>
```

## Supporting and Managing Arabic Domain Names



```
variant"/>
    <var cp="0671" type="block" when="arabic-final-right-join" comment="Typo variant"/>
    <var cp="0671" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
    <var cp="0672" type="block" when="arabic-final-right-join" comment="Typo variant"/>
    <var cp="0672" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
    <var cp="0673" type="block" when="arabic-final-right-join" comment="Typo variant"/>
    <var cp="0673" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
    <var cp="0675" type="block" when="arabic-final-right-join" comment="Typo variant"/>
    <var cp="0675" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
    <var cp="0773" type="block" when="arabic-final-right-join" comment="Typo variant"/>
    <var cp="0773" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
    <var cp="0774" type="block" when="arabic-final-right-join" comment="Typo variant"/>
    <var cp="0774" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
</char>

<char cp="0628" ref="0 1 2" />

<char cp="0629" ref="0 1 2">
    <var cp="0647" type="allocate" when="arabic-final-dual-join" comment="Language variant"/>
    <var cp="0647" type="allocate" when="arabic-isolated-dual-join" comment="Language
variant"/>
    <var cp="06BE" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
    <var cp="06BE" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
    <var cp="06C1" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
    <var cp="06C1" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
    <var cp="06C3" type="activate" when="arabic-final-right-join" comment="Typo variant"/>
    <var cp="06C3" type="activate" when="arabic-isolated-right-join" comment="Exact variant"/>
    <var cp="06D5" type="block" when="arabic-final-right-join" comment="Typo variant"/>
    <var cp="06D5" type="block" when="arabic-isolated-right-join" comment="Typo variant"/>
</char>

<char cp="062A" ref="0 1 2">
    <var cp="063E" type="block" when="arabic-initial-dual-join" comment="Exact variant"/>
    <var cp="063E" type="block" when="arabic-medial-dual-join" comment="Exact variant"/>
    <var cp="067A" type="block" comment="Typo variant"/>
</char>

<char cp="062B" ref="0 1 2">
    <var cp="063F" type="block" when="arabic-initial-dual-join" comment="Exact variant"/>
    <var cp="063F" type="block" when="arabic-medial-dual-join" comment="Exact variant"/>
    <var cp="067D" type="block" comment="Typo variant"/>
    <var cp="06BD" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
    <var cp="06BD" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
</char>

<char cp="062C" ref="0 1 2" />

<char cp="062D" ref="0 1 2" />

<char cp="062E" ref="0 1 2" />

<char cp="062F" ref="0 1 2" />
```

## Supporting and Managing Arabic Domain Names



```
<char cp="0630" ref="0 1 2" />

<char cp="0631" ref="0 1 2" />

<char cp="0632" ref="0 1 2" />

<char cp="0633" ref="0 1 2" />

<char cp="0634" ref="0 1 2" />

<char cp="0635" ref="0 1 2" />

<char cp="0636" ref="0 1 2" />

<char cp="0637" ref="0 1 2" />

<char cp="0638" ref="0 1 2" />

<char cp="0639" ref="0 1 2" />

<char cp="063A" ref="0 1 2" />

<char cp="0641" ref="0 1 2">
  <var cp="06A7" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
  <var cp="06A7" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
  <var cp="06A7" type="block" when="arabic-initial-dual-join" comment="Exact variant"/>
  <var cp="06A7" type="block" when="arabic-medial-dual-join" comment="Exact variant"/>
</char>

<char cp="0642" ref="0 1 2" />

<char cp="0643" ref="0 1 2">
  <var cp="06A9" type="activate" when="arabic-final-dual-join" comment="Typo variant"/>
  <var cp="06A9" type="activate" when="arabic-isolated-dual-join" comment="Typo variant"/>
  <var cp="06A9" type="activate" when="arabic-initial-dual-join" comment="Exact variant"/>
  <var cp="06A9" type="activate" when="arabic-medial-dual-join" comment="Exact variant"/>
  <var cp="06AA" type="block" comment="Typo variant"/>
</char>

<char cp="0644" ref="0 1 2" />

<char cp="0645" ref="0 1 2" />

<char cp="0646" ref="0 1 2">
  <var cp="06BA" type="block" when="arabic-initial-dual-join" comment="Exact variant"/>
  <var cp="06BA" type="block" when="arabic-medial-dual-join" comment="Exact variant"/>
</char>

<char cp="0647" ref="0 1 2">
  <var cp="0629" type="allocate" when="arabic-final-dual-join" comment="Language variant"/>
  <var cp="0629" type="allocate" when="arabic-isolated-dual-join" comment="Language
variant"/>
  <var cp="06BE" type="activate" when="arabic-initial-dual-join" comment="Exact variant"/>
```

## Supporting and Managing Arabic Domain Names



```
<var cp="06BE" type="activate" when="arabic-medial-dual-join" comment="Exact variant"/>
<var cp="06BE" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
<var cp="06BE" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
<var cp="06C1" type="activate" when="arabic-isolated-dual-join" comment="Exact variant"/>
<var cp="06C1" type="activate" when="arabic-final-dual-join" comment="Typo variant"/>
<var cp="06C1" type="block" when="arabic-initial-dual-join" comment="Typo variant"/>
<var cp="06C1" type="block" when="arabic-medial-dual-join" comment="Typo variant"/>
<var cp="06C3" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
<var cp="06C3" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
<var cp="06D5" type="block" when="arabic-final-dual-join" comment="Exact variant"/>
<var cp="06D5" type="block" when="arabic-isolated-dual-join" comment="Exact variant"/>
</char>

<char cp="0648" ref="0 1 2" />

<char cp="0649" ref="0 1 2">
  <var cp="064A" type="allocate" when="arabic-final-dual-join" comment="Language variant"/>
  <var cp="064A" type="allocate" when="arabic-isolated-dual-join" comment="Language
variant"/>

  <var cp="066E" type="block" when="arabic-initial-dual-join" comment="Exact variant"/>
  <var cp="066E" type="block" when="arabic-medial-dual-join" comment="Exact variant"/>
  <var cp="06CC" type="activate" when="arabic-final-dual-join" comment="Exact variant"/>
  <var cp="06CC" type="activate" when="arabic-isolated-dual-join" comment="Exact variant"/>
  <var cp="06CD" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
  <var cp="06CD" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
  <var cp="06D2" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
  <var cp="06D2" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
</char>

<char cp="064A" ref="0 1 2">
  <var cp="0649" type="allocate" when="arabic-final-dual-join" comment="Language variant"/>
  <var cp="0649" type="allocate" when="arabic-isolated-dual-join" comment="Language
variant"/>

  <var cp="067B" type="block" when="arabic-initial-dual-join" comment="Typo variant"/>
  <var cp="067B" type="block" when="arabic-medial-dual-join" comment="Typo variant"/>
  <var cp="06CC" type="activate" when="arabic-initial-dual-join" comment="Exact variant"/>
  <var cp="06CC" type="activate" when="arabic-medial-dual-join" comment="Exact variant"/>
  <var cp="06CC" type="activate" when="arabic-isolated-dual-join" comment="Typo variant"/>
  <var cp="06CC" type="activate" when="arabic-final-dual-join" comment="Typo variant"/>
  <var cp="06CD" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
  <var cp="06CD" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
  <var cp="06D0" type="block" comment="Typo variant"/>
  <var cp="06D2" type="block" when="arabic-final-dual-join" comment="Typo variant"/>
  <var cp="06D2" type="block" when="arabic-isolated-dual-join" comment="Typo variant"/>
</char>

<char cp="0660" ref="0 1 2">
  <var cp="0030" type="activate" comment="Typo variant"/>
  <var cp="06F0" type="activate" comment="Exact variant"/>
</char>

<char cp="0661" ref="0 1 2">
  <var cp="0031" type="activate" comment="Typo variant"/>
```

## Supporting and Managing Arabic Domain Names



```
<var cp="06F1" type="activate" comment="Exact variant"/>
</char>

<char cp="0662" ref="0 1 2">
  <var cp="0032" type="activate" comment="Typo variant"/>
  <var cp="06F2" type="activate" comment="Exact variant"/>
</char>

<char cp="0663" ref="0 1 2">
  <var cp="0033" type="activate" comment="Typo variant"/>
  <var cp="06F3" type="activate" comment="Exact variant"/>
</char>

<char cp="0664" ref="0 1 2">
  <var cp="0034" type="activate" comment="Typo variant"/>
  <var cp="06F4" type="activate" comment="Typo variant"/>
</char>

<char cp="0665" ref="0 1 2">
  <var cp="0035" type="activate" comment="Typo variant"/>
  <var cp="06F5" type="activate" comment="Typo variant"/>
</char>

<char cp="0666" ref="0 1 2">
  <var cp="0036" type="activate" comment="Typo variant"/>
  <var cp="06F6" type="activate" comment="Typo variant"/>
</char>

<char cp="0667" ref="0 1 2">
  <var cp="0037" type="activate" comment="Typo variant"/>
  <var cp="06F7" type="activate" comment="Exact variant"/>
</char>

<char cp="0668" ref="0 1 2">
  <var cp="0038" type="activate" comment="Typo variant"/>
  <var cp="06F8" type="activate" comment="Exact variant"/>
</char>

<char cp="0669" ref="0 1 2">
  <var cp="0039" type="activate" comment="Typo variant"/>
  <var cp="06F9" type="activate" comment="Exact variant"/>
</char>

<char cp="0030" ref="0 1 2">
  <var cp="0660" type="activate" comment="Typo variant"/>
  <var cp="06F0" type="activate" comment="Typo variant"/>
</char>

<char cp="0031" ref="0 1 2">
  <var cp="0661" type="activate" comment="Typo variant"/>
  <var cp="06F1" type="activate" comment="Typo variant"/>
</char>
```

## Supporting and Managing Arabic Domain Names



```
<char cp="0032" ref="0 1 2">
  <var cp="0662" type="activate" comment="Typo variant"/>
  <var cp="06F2" type="activate" comment="Typo variant"/>
</char>

<char cp="0033" ref="0 1 2">
  <var cp="0663" type="activate" comment="Typo variant"/>
  <var cp="06F3" type="activate" comment="Typo variant"/>
</char>

<char cp="0034" ref="0 1 2">
  <var cp="0664" type="activate" comment="Typo variant"/>
  <var cp="06F4" type="activate" comment="Typo variant"/>
</char>

<char cp="0035" ref="0 1 2">
  <var cp="0665" type="activate" comment="Typo variant"/>
  <var cp="06F5" type="activate" comment="Typo variant"/>
</char>

<char cp="0036" ref="0 1 2">
  <var cp="0666" type="activate" comment="Typo variant"/>
  <var cp="06F6" type="activate" comment="Typo variant"/>
</char>

<char cp="0037" ref="0 1 2">
  <var cp="0667" type="activate" comment="Typo variant"/>
  <var cp="06F7" type="activate" comment="Typo variant"/>
</char>

<char cp="0038" ref="0 1 2">
  <var cp="0668" type="activate" comment="Typo variant"/>
  <var cp="06F8" type="activate" comment="Typo variant"/>
</char>

<char cp="0039" ref="0 1 2">
  <var cp="0669" type="activate" comment="Typo variant"/>
  <var cp="06F9" type="activate" comment="Typo variant"/>
</char>

<char cp="002D" ref="0 1 2" />
</data>
<rules>

  <class name="arabic-language">0621 0622 0623 0624 0625 0626 0627 0628 0629 062A 062B 062C 062D
062E 062F 0630 0631 0632 0633 0634 0635 0636 0637 0638 0639 063A 0641 0642 0643 0644 0645 0646 0647 0648 0649 064A
0660 0661 0662 0663 0664 0665 0666 0667 0668 0669 0030 0031 0032 0033 0034 0035 0036 0037 0038 0039 002D</class>
  <class name="persian-language">0621 0622 0623 0624 0626 0627 0628 0629 062A 062B 062C 062D 062E
062F 0630 0631 0632 0633 0634 0635 0636 0637 0638 0639 063A 0641 0642 0644 0645 0646 0647 0648 067E 0686 0698 06A9
06AF 06CC 06F0 06F1 06F2 06F3 06F4 06F5 06F6 06F7 06F8 06F9 002D 0030 0031 0032 0033 0034 0035 0036 0037 0038
0039</class>
  <class name="urdu-language">0621 0622 0623 0624 0627 0628 062A 062B 062C 062D 062E 062F 0630 0631
0632 0633 0634 0635 0636 0637 0638 0639 063A 0641 0642 0644 0645 0646 0648 0679 067E 0686 0688 0691 0698 06A9 06AF
```

## Supporting and Managing Arabic Domain Names



```
06BA 06BE 06C1 06C2 06C3 06CC 06D2 06D3 06F0 06F1 06F2 06F3 06F4 06F5 06F6 06F7 06F8 06F9 0030 0031 0032 0033
0034 0035 0036 0037 0038 0039 002D</class>
    <class name="malay-language">0621 0623 0625 0626 0627 0628 0629 062A 062B 062C 062D 062E 062F
0630 0631 0632 0633 0634 0635 0636 0637 0638 0639 063A 0641 0642 0644 0645 0646 0647 0648 0649 064A 0686 06A0 06A4
06A9 06BD 06CF 06F2 0762 0030 0031 0032 0033 0034 0035 0036 0037 0038 0039 002D</class>
    <class name="pashto-language">0621 0622 0623 0624 0626 0627 0628 0629 062A 062B 062C 062D 062E
062F 0630 0631 0632 0633 0634 0635 0636 0637 0638 0639 063A 0641 0642 0644 0645 0646 0647 0648 064A 0660 0661 0662
0663 0664 0665 0666 0667 0668 0669 067C 067E 0685 0686 0689 0693 0696 0698 069A 06A9 06AF 06BC 06C0 06CC 06CD
06D0 0030 0031 0032 0033 0034 0035 0036 0037 0038 0039 002D</class>

    <union name="supported-code-points">
        <class by-ref="arabic-language"/>
        <class by-ref="persian-language"/>
        <class by-ref="urdu-language"/>
        <class by-ref="malay-language"/>
        <class by-ref="pashto-language"/>
    </union>

    <union name="digit-set">
        <class by-ref="arabic-digits"/>
        <class by-ref="arabic-indic-digits"/>
        <class by-ref="extended-arabic-indic-digits"/>
    </union>

    <difference name="supported-code-points-without-digits">
        <class by-ref="supported-code-points"/>
        <class by-ref="digit-set"/>
    </difference>

    <class name="transparent" property="jt:T"/>
    <class name="right-joining" property="jt:R"/>
    <class name="left-joining" property="jt:L"/>
    <class name="dual-joining" property="jt:D"/>
    <class name="non-joining" property="jt:U"/>
    <class name="join-causing" property="jt:C"/>

    <class name="arabic-script" property="sc:Arab"/>
    <class name="arabic-digits">0030-0039</class>
    <class name="arabic-indic-digits">0660-0669</class>
    <class name="extended-arabic-indic-digits">06F0-06F9</class>

    <class name="word-separator">002D</class>

    <rule name="no-mixing-script" comment="only labels from Arabic script is allowed">
        <start />
        <union count="1+">
            <class by-ref="arabic-script"/>
            <class by-ref="arabic-digits"/>
            <class by-ref="arabic-indic-digits"/>
            <class by-ref="extended-arabic-indic-digits"/>
            <class by-ref="word-separator"/>
        </union>
```

## Supporting and Managing Arabic Domain Names



```
        <end />
</rule>

<rule name="no-consecutive-hyphen-rule">
    <any/>
    <char cp="002D" comment="literal HYPHEN"/>
    <char cp="002D" comment="literal HYPHEN"/>
    <any/>
</rule>

<rule name="no-hyphen-at-start">
    <start/>
    <char cp="002D" comment="literal HYPHEN"/>
    <any/>
</rule>

<rule name="no-hyphen-at-end">
    <any/>
    <char cp="002D" comment="literal HYPHEN"/>
    <end/>
</rule>

<rule name="no-digit-at-start" comment="digits at the beginning is disallowed">
    <start/>
    <choice count="1+">
        <class by-ref="arabic-digits"/>
        <class by-ref="arabic-indic-digits"/>
        <class by-ref="extended-arabic-indic-digits"/>
    </choice>
    <class by-ref="supported-code-points-without-digits"/>
    <any count="0+"/>
</rule>

<rule name="no-digit-mixing">
    <choice count="1+">
        <rule>
            <start/>
            <any count="0+"/>
            <class by-ref="arabic-digits"/>
            <any count="0+"/>
            <choice count="1+">
                <class by-ref="arabic-indic-digits"/>
                <class by-ref="extended-arabic-indic-digits"/>
            </choice>
            <any count="0+"/>
            <end/>
        </rule>

        <rule>
            <start/>
            <any count="0+"/>
            <class by-ref="arabic-indic-digits"/>
            <any count="0+"/>
        </rule>
    </choice>
</rule>
```

## Supporting and Managing Arabic Domain Names



```

        <choice count="1+">
            <class by-ref="arabic-digits"/>
            <class by-ref="extended-arabic-indic-digits"/>
        </choice>
        <any count="0+"/>
        <end/>
    </rule>
</rule>
    <start/>
    <any count="0+"/>
    <class by-ref="extended-arabic-indic-digits"/>
    <any count="0+"/>
    <choice count="1+">
        <class by-ref="arabic-digits"/>
        <class by-ref="arabic-indic-digits"/>
    </choice>
    <any count="0+"/>
    <end/>
</rule>
</choice>
</rule>
<rule name="no-mixing-languages" comment="a label can only be written using characters from one of
the 5 languages">
    <choice count="1+">
        <rule>
            <start />
            <class by-ref="arabic-language" count="1+"/>
            <end />
        </rule>
        <rule>
            <start />
            <class by-ref="persian-language" count="1+"/>
            <end />
        </rule>
        <rule>
            <start />
            <class by-ref="urdu-language" count="1+"/>
            <end />
        </rule>
        <rule>
            <start />
            <class by-ref="pashto-language" count="1+"/>
            <end />
        </rule>
        <rule>
            <start />
            <class by-ref="malay-language" count="1+"/>
            <end />
        </rule>
    </choice>
</rule>

```

## Supporting and Managing Arabic Domain Names



```
<rule name="no-connected-alef-maksura" comment="a label which has connected alef maksura is
blocked">
    <start/>
    <any count="0+"/>
    <char cp="0649" count="1+"/>
    <choice count="1+">
        <class by-ref="right-joining" />
        <class by-ref="dual-joining" />
    </choice>
    <any count="0+"/>
    <end/>
</rule>

<rule name="no-alef-with-hamzah-or-mada-at-end" comment="a label which has alef hamzah or mada
at the end is blocked">
    <start/>
    <any count="0+"/>
    <choice count="1+">
        <class by-ref="left-joining" />
        <class by-ref="dual-joining" />
    </choice>
    <choice count="1+">
        <char cp="0625" count="1+"/>
        <char cp="0622" count="1+"/>
    </choice>
    <choice count="1+">
        <char cp="002D" comment="literal HYPHEN"/>
    <end/>
    </choice>
</rule>

<rule name="arabic-initial-dual-join">
    <look-behind>
        <choice count="1+">
            <start/>
            <rule>
                <class by-ref="transparent" count="0+"/>
                <class by-ref="non-joining"/>
            </rule>
            <rule>
                <class by-ref="transparent" count="0+"/>
                <char cp="002D" comment="literal HYPHEN"/>
            </rule>
            <rule>
                <class by-ref="transparent" count="0+"/>
                <char cp="0629" comment="TEH MARBUTA"/>
            </rule>
            <rule>
                <class by-ref="transparent" count="0+"/>
                <class by-ref="right-joining"/>
            </rule>
        </choice>
```

## Supporting and Managing Arabic Domain Names



```
</look-behind>
<anchor/>
<look-ahead>
  <class by-ref="transparent" count="0+"/>
  <choice count="1+ ">
    <class by-ref="right-joining" />
    <class by-ref="dual-joining" />
  </choice>
</look-ahead>
</rule>

<rule name="arabic-medial-dual-join">
  <look-behind>
    <choice count="1+ ">
      <rule>
        <class by-ref="transparent" count="0+"/>
        <class by-ref="dual-joining"/>
      </rule>
      <rule>
        <class by-ref="transparent" count="0+"/>
        <class by-ref="left-joining"/>
      </rule>
    </choice>
  </look-behind>
  <anchor/>
  <look-ahead>
    <class by-ref="transparent" count="0+"/>
    <choice count="1+ ">
      <class by-ref="right-joining" />
      <class by-ref="dual-joining" />
    </choice>
  </look-ahead>
</rule>

<rule name="arabic-final-dual-join">
  <look-behind>
    <choice count="1+ ">
      <rule>
        <class by-ref="transparent" count="0+"/>
        <class by-ref="left-joining"/>
      </rule>
      <rule>
        <class by-ref="transparent" count="0+"/>
        <class by-ref="dual-joining"/>
      </rule>
    </choice>
  </look-behind>
  <anchor/>
  <look-ahead>
    <choice count="1+ ">
      <rule>
        <class by-ref="transparent" count="0+"/>
        <class by-ref="non-joining"/>
      </rule>
    </choice>
  </look-ahead>
</rule>
```

## Supporting and Managing Arabic Domain Names



```

        </rule>
        <rule>
            <class by-ref="transparent" count="0+"/>
            <char cp="002D" comment="literal HYPHEN"/>
        </rule>
    </end/>
</choice>
</look-ahead>
</rule>
<rule name="arabic-isolated-dual-join">
    <look-behind>
        <choice count="1+">
            <start/>
            <rule>
                <class by-ref="transparent" count="0+"/>
                <class by-ref="non-joining"/>
            </rule>
            <rule>
                <class by-ref="transparent" count="0+"/>
                <class by-ref="right-joining"/>
            </rule>
            <rule>
                <class by-ref="transparent" count="0+"/>
                <char cp="002D" comment="literal HYPHEN"/>
            </rule>
        </choice>
    </look-behind>
    <anchor/>
    <look-ahead>
        <choice count="1+">
            <rule>
                <class by-ref="transparent" count="0+"/>
                <class by-ref="non-joining"/>
            </rule>
            <rule>
                <class by-ref="transparent" count="0+"/>
                <class by-ref="left-joining"/>
            </rule>
            <rule>
                <class by-ref="transparent" count="0+"/>
                <char cp="002D" comment="literal HYPHEN"/>
            </rule>
        </choice>
    </look-ahead>
</rule>

<rule name="arabic-isolated-right-join">
    <look-behind>
        <choice count="1+">
            <start/>
            <rule>
                <class by-ref="transparent" count="0+"/>
            </rule>
        </choice>
    </look-behind>
    <look-ahead>
        <choice count="1+">
            <start/>
            <rule>
                <class by-ref="transparent" count="0+"/>
            </rule>
        </choice>
    </look-ahead>
</rule>

```

## Supporting and Managing Arabic Domain Names



```

                                <class by-ref="non-joining"/>
                                </rule>
                                <rule>
                                <class by-ref="transparent" count="0+"/>
                                <char cp="002D" comment="literal HYPHEN"/>
                                </rule>
                                <rule>
                                <class by-ref="transparent" count="0+"/>
                                <char cp="0629" comment="TEH MARBUTA"/>
                                </rule>
                                <rule>
                                <class by-ref="transparent" count="0+"/>
                                <class by-ref="right-joining"/>
                                </rule>
                                </choice>
                                </look-behind>
                                <anchor/>
                                <look-ahead>
                                <any count="0+"/>
                                </look-ahead>
                                </rule>

                                <rule name="arabic-final-right-join">
                                <look-behind>
                                <choice count="1+">
                                <class by-ref="left-joining" />
                                <class by-ref="dual-joining" />
                                </choice>
                                </look-behind>
                                <anchor/>
                                <look-ahead>
                                <any count="0+"/>
                                </look-ahead>
                                </rule>

                                <action disp="invalid" not-match="no-mixing-script" />
                                <action disp="invalid" not-match="no-mixing-languages" />
                                <action disp="invalid" match="no-hyphen-at-start" />
                                <action disp="invalid" match="no-hyphen-at-end" />
                                <action disp="invalid" match="no-consecutive-hyphen-rule" />
                                <action disp="invalid" match="no-connected-alef-maksura" />
                                <action disp="block" match="no-alef-with-hamzah-or-mada-at-end" />
                                <action disp="invalid" match="no-digit-at-start" />
                                <action disp="invalid" match="no-digit-mixing" />
                                </rules>
                                </lgr>
```

Latest version can be found at: <https://www.iana.org/domains/idn-tables>

## Appendix 3: Guideline Rules for writing Arabic Labels under (.السعودية)

This section sets a number of guideline rules that help to write Arabic domain names correctly, taking into account issues related to the Arabic language as well as the Arabic script. ( [http://nic.sa/en/view/writing\\_arabic\\_idn\\_guideline](http://nic.sa/en/view/writing_arabic_idn_guideline) )

The terms and expressions used in this section shall have the same definitions and meanings as in the Saudi Domain Names Registration Regulation document.

Recommendations outlined in the RFC 5564 entitled: "Linguistic Guidelines for the Use of the Arabic Language in Internet Domains" will be followed. They are in line with the specifications put together and agreed upon by the Arab Team for domain names and Internet affairs, which operates under the umbrella of the League of Arab States.

It is an Arabic specification that SaudiNIC had participated majorly in drafting. It contains some recommendations for linguistic issues related to Arabic domain names and its language table. It is available on the following link: <http://tools.ietf.org/html/rfc5564>

### Rule 1:

- Diacritics are not allowed , Examples :  
سَجَل.السعودية Not Accepted  
سجل.السعودية Accepted

### Rule 2:

- No mixing between scripts (Arabic & Latin), Examples :  
موقع-SaudiNIC.السعودية Not Accepted  
SaudiNIC.السعودية Not Accepted

### Rule 3:

- Use of hyphen (instead of space) between words particularly if the 2 words will get connected , Examples :  
هيئةالاتصالات.السعودية Accepted  
هيئة-الاتصالات.السعودية Accepted  
مدارس-خيف.السعودية Accepted  
مدارسخيف.السعودية Not Accepted

### Rule 4:

- In the Arab language words are normally separated by spaces. Connecting words without spaces is usually not acceptable and it may decrease readability. It has been recommended that multiple words are separated by the character "-" dash.



- As the "dash" is not typically used in Arabic texts and it is imposed on the users, and for the purpose of simplifying domain name variants management by registrants and to increase security and stability by reducing useless and impracticable cases, the usage of "dash" in Arabic IDNs will be limited to the following conditions:
  - Hyphen cannot be used at the beginning or end of a label, and 2 or more consecutive hyphens are not allowed, Examples :
    - هئيةالات.السعودية Not Accepted
    - هئيةالات.-السعودية Not Accepted
    - هئية--الاتصالات.السعودية Not Accepted
  - Hyphen must be used between words particularly if the two words will be connected otherwise it will be considered different label. For example, the following are considered different domain names:
    - مدارسخيف.السعودية Not Accepted
    - مدارس-خيف.السعودية Accepted
  - For the case of words that will not be connected, both forms (with or without a "dash") will be considered variants. For example, the following will be considered as variant domain names:
    - مدرسةحنين.السعودية Accepted
    - مدرسة-حنين.السعودية Accepted

### Rule 5:

- Digits cannot be used at the beginning or end of a label Digits can be used inside a label from the 2 sets Arabic-Indic digits (٠،١،٢،٣،٤،٥،٦،٧،٨،٩) and Arabic digits (0,1,2,3,4,5,6,7,8,9) without mixing, Examples :
  - ٩٩٩.السعودية Not Accepted
  - اتصل ٩٩٩.السعودية Not Accepted
  - ٩٩ وللنجدة.السعودية Not Accepted
  - اتصل ٩٩٩ للنجدة.السعودية Not Accepted
  - اتصل ٩٩٩ للنجدة.السعودية Accepted
- It is permitted to register other variants of the domain name that can be formulated just by changing the digit set , Examples :
  - اتصل ٩٩٩ للنجدة.السعودية Accepted

### Rule 6:

- It is permitted to register the Variants within the Language After registering a domain name that contains one or more of the confusingly similar characters, e.g. :
  - ALEF MAKSURA and YEH only at the end of a word
  - ALEF forms:
    - If the domain name contains "ALEF WITH MADDA ABOVE", "ALEF WITH HAMZA ABOVE", or "ALEF WITH HAMZA BELOW" then you can have a variant with ALEF and vice versa.
    - If the domain name contains "ALEF WITH MADDA ABOVE" then you can have a variant with "ALEF WITH HAMZA ABOVE" and vice versa.
  - TEH MARBUTA and TEH at the end of word.



- It is permitted to register the other variants names that can be created just by changing the confusingly similar characters provided that it does not infringe on the rights of others, Examples :  
شبكة-الأخبار.السعودية **Accepted**
- It is permitted to register the other variants taking care of the consequences of the abovementioned condition, Examples :  
شبكة-الاخبار.السعودية **Accepted with condition**  
شبكة-الأخبار.السعودية **Accepted with condition**  
شبكة-الاخبار.السعودية **Accepted with condition**

### Rule 7:

- It is permitted to register the Variants within the Arabic Script that will allow the Arabic domain names to be used globally despite the fact that some other languages that use the Arabic script (such as Urdu, Farsi, Pashto, ...) might have characters that look confusingly similar to some of the Arabic characters. This also takes care of the problem of having different input devices (e.g., keyboards) used within the Arabic script communities. SaudiNIC will provide an appropriate mechanism for dealing with domain names variants and it will enable access to the Arabic domain names even if they contained confusingly similar characters from languages that uses Arabic script (such as Farsi and Urdu). Examples :  
مكة.السعودية **Accepted** (all characters are from the Arabic Language)
- It is permitted to register the other variants, Examples :  
مكة.السعودية **Accepted** (the letter KAF and THE MARBUTA are from Urdu)
- Also a 3<sup>rd</sup> set of numbers "Eastern Arabic-Indic digits" (٠,١,٢,٣,٤,٥,٦,٧,٨,٩) will be added to Rule 5 and they should not mixed with others sets.

## Appendix 4: Control Characters: ZWJ and ZWNJ

### ZWNJ Confusing Cases

This section lists some cases when the ZWNJ control character has no effect on the shape of the character it follows and hence it cause confusion with the same character sequence when ZWNJ is not used.

### Usage of ZWNJ

The Unicode Standard provides a user-selectable formatting codes: U+200C zero width non-joiner and U+200D zero width joiner. The use of a non-joiner between two letters prevents those letters from forming a cursive connection with each other when rendered. Examples include the Persian plural suffix, some Persian proper names, and Ottoman Turkish vowels. The Unicode Standard 5.0 - Chapter 8 Middle Eastern Scripts.

Format (sequence of typing ... right-to-left)

<Succeeding Character> ZWNJ <Preceding Character>

SChar = Succeeding Character , PChar = Preceding Character>

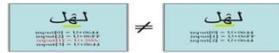
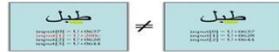
1. When the shapes of PChar with or without left joining are similar

This is valid for characters with the TAH shape (ط) which include the following characters:

- TAH U+0637 ط
- ZAH U+0638 ظ
- TAH with three dots above U+069f ظ
- U+06BE
- U+06FF
- U+08A3
- 0637
- 0638
- 069F

Examples not using ZWNJ	Examples not using ZWNJ
طبل	طبل
input[0] = U+0637	input[0] = U+0637
input[1] = U+0628	input[1] = U+200c
input[2] = U+0644	input[2] = U+0628
	input[3] = U+0644

## Supporting and Managing Arabic Domain Names



<http://demo.icu-project.org/icu-bin/idnbrowser?t=%D8%B7%E2%80%8C%D8%A8%D9%84>  
<http://unicode.org/cldr/utility/idna.jsp?a=%D8%B7%D8%A8%D9%84%0D%0A%D8%B7%E2%80%8C%D8%A8%D9%84>

### 2. When SChar is a digit

Since digits are not right joining characters, then having ZWNJ before them has no effect.

- European U+0030..U+0039 (0123456789)
- Arabic-Indic U+0660..U+0669 (٠١٢٣٤٥٦٧٨٩)
- Eastern Arabic-Indic U+06F0..U+06F9 (٠١٢٣٤٥٦٧٨٩)

Examples not using ZWNJ	Examples not using ZWNJ
ب١	ب١
input[0] = U+0628 input[2] = U+0661	input[0] = U+0628 input[1] = U+200c input[2] = U+0661

### 3. When SChar is a dash or a dot

## Supporting and Managing Arabic Domain Names



“Dash” and dot (used as label separator) are not right joining characters, having ZWNJ before them has no effect.

- Dash U+002d
- Dot U+002e

Examples not using ZWNJ	Examples not using ZWNJ
-ب	ب
input[0] = U+0628 input[2] = U+002d	input[0] = U+0628 <span style="color: red;">input[1] = U+200c</span> input[3] = U+002d

Examples not using ZWNJ	Examples not using ZWNJ
.ب	ب.
input[0] = U+0628 input[3] = U+002e	input[0] = U+0628 <span style="color: red;">input[1] = U+200c</span> input[3] = U+002e

#### 4. When SChar is Hamza and High Hamza

- Hamza and High Hamza are not right joining characters. Therefore, having ZWNJ before them has no effect.
- Hamza U+0621
- High Hamza U+0674

Examples not using ZWNJ	Examples not using ZWNJ
بء	ءب
input[0] = U+0628 input[2] = U+0621	input[0] = U+0628 <span style="color: red;">input[1] = U+200c</span> input[2] = U+0621

Examples not using ZWNJ	Examples not using ZWNJ
بْ	بُ
input[0] = U+0628 input[2] = U+0674	input[0] = U+0628 <span style="color: red;">input[1] = U+200c</span> input[2] = U+0674

#### 5. When SChar is ALF Maqsoorah

These control characters are not known for the Arab Communities, however they might be known for other communities (such as Persian and Urdu). SaudiNIC highly recommends not supporting these control characters. However, if a registry decided to support them they should be very careful for the security issues that may raise.