VXML 2.1 TUTORIAL pdf

1: Introduction to CCXML, Part II - www.enganchecubano.com

VoiceXML is an extensible markup language (XML) for the creation of automated speech recognition (ASR) and interactive voice response (IVR) applications.

The version of VoiceXML of this document required. The current version number is 2. The namespace for VoiceXML is defined to be http: If omitted, the value is a platform-specific default. Language information is inherited down the document hierarchy: In cases where you want multiple documents to work together as one application, you select one document to be the application root document, and the rest to be application leaf documents. When this is done, every time the interpreter is told to load and execute a leaf document in this application, it first loads the application root document if it is not already loaded. The application root document remains loaded until the interpreter is told to load a document that belongs to a different application. Thus one of the following two conditions always holds during interpretation: The application root document is loaded and the user is executing in it: The application root document and a single leaf document are both loaded and the user is executing in the leaf document. If there is a chain of subdialogs defined in separate documents, then there may be more than one leaf document loaded although execution will only be in one of these documents. When a leaf document load causes a root document load, none of the dialogs in the root document are executed. Execution begins in the leaf document. There are several benefits to multi-document applications. Document-scoped grammars in the root document are active when the user is in a leaf document, so that the user is able to interact with forms, links, and menus in the root document. Here is a two-document application illustrating this: Application root document app-root. Its application attribute specifies that app-root. Shall we say Ciao? I did not understand what you said. Note that when the user is in a multi-document application, at most two documents are loaded at any one time: An interpreter always has an application root document loaded; it does not always have an application leaf document loaded. The absolute URI includes a query string, if present, but it does not include a fragment identifier. The interpreter remains in the same application as long as the name remains the same. When the name changes, a new application is entered and its root context is initialized. Some transitions are within an application, others are between applications. The preservation or initialization of the root context depends on the type of transition: The application root document and its context are preserved. Root to Root A root to root transition occurs when the current document is a root document and the target document is a root document, i. The root context is initialized with the application root document returned by the caching policy in Section 6. The caching policy is consulted even when the name of the target application and the current application are the same. As discussed in Section 2. When the subdialog is invoked with a non-empty URI reference, the caching policy in Section 6. If a subdialog is invoked with an empty URI reference and a fragment identifier, e. If a document refers to a non-existent application root document, an error. The following diagrams illustrate the effect of the transitions between root and leaf documents on the application root context. Transitions that Preserve the Root Context In this diagram, all the documents belong to the same application. The transitions are identified by the numbers across the top of the figure. Assume that this is the first document in the session. Document 1 specifies a transition to URI B, which yields document 2. The root is document 1 with its context preserved. This is a root to leaf transition within the same application. Document 2 specifies a transition to URI C, which yields another leaf document, document 3. Its application attribute also equals URI A. This is a leaf to leaf transition within the same application. Document 1 is used with its root context intact. This is a leaf to root transition within the same application. The next diagram illustrates transitions which initialize the root context. The resulting document 4 does not have an application attribute, so it is considered a root document, and the root context is initialized. This is a root to root transition. Document 4 specifies a transition to URI D, which yields a leaf document 5. Its application attribute is different: A new application is being entered. URI E produces the root document 6. The root context is initialized from the content of document 6. This is an inter-application transition. Document 5 specifies a transition to URI A. The cache check returns document 4 which does not have an application attribute and therefore belongs to application A, so the root context is

initialized. Initialization occurs even though this application and this root document were used earlier in the session. For example, the solicitation of account information may involve gathering several pieces of information, such as account number, and home telephone number. A customer care service might be structured with several independent applications that could share this basic building block, thus it would be reasonable to construct it as a subdialog. This is illustrated in the example below. The first document, app. The account information is obtained by using a subdialog element that invokes another VoiceXML document to solicit the user input. While the second document is being executed, the calling dialog is suspended, awaiting the return of information. Customer Service Application app. The subdialog could be a new dialog within the existing document, or a new dialog within a new document. Subdialogs can be composed of several documents. Figure 5 shows the execution flow where a sequence of documents D transitions to a subdialog SD and then back. Subdialog composed of several documents returning from the last subdialog document. The execution context in dialog D2 is suspended when it invokes the subdialog SD1 in document sd1. This subdialog specifies execution is to be transfered to the dialog in sd2. Consequently, when the dialog in sd2. Figure 6 shows an example of a multi-document subdialog where control is transferred from one subdialog to another. Subdialog composed of several documents returning from the first subdialog document. The subdialog in sd1. When executing SD2, there are two suspended contexts: When SD2 returns, control is returned to the SD1. It in turn returns control to dialog D2. The purpose of this state is to allow the VoiceXML application to perform any necessary final cleanup, such as submitting information to the application server. Aside from this restriction, execution of the VoiceXML application continues normally while in the final processing state. Thus for example the application may transition between documents while in the final processing state, and the interpreter must exit if no form item is eligible to be selected as described in Section 2. A set of form items, elements that are visited in the main loop of the form interpretation algorithm. Declarations of non-form item variables. If specified, the form can be referenced within the document or from another document. If it is dialog then the form grammars are active only in the form. If the scope is document, then the form grammars are active during any dialog in the same document. If the scope is document and the document is an application root document, then the form grammars are active during any dialog in any document of this application. Note that the scope of individual form grammars takes precedence over the default scope; for example, in non-root documents a form with the default scope "dialog", and a form grammar with the scope "document", then that grammar is active in any dialog in the document. This section describes some of the concepts behind forms, and then gives some detailed examples of their operation. The FIA has a main loop that repeatedly selects a form item and then visits it. The selected form item is the first in document order whose guard condition is not satisfied. Interpreting a form item generally involves: The FIA ends when it interprets a transfer of control statement e. The FIA is described in more detail in Section 2. Input items direct the FIA to gather a result for a specific element. When the FIA selects a control item, the control item may contain a block of procedural code to execute, or it may tell the FIA to set up the initial prompt-and-collect for a mixed initiative form. Input items have prompts to tell the user what to say or key in, grammars that define the allowed inputs, and event handlers that process any resulting events.

2: Dave Raggetts Introduction to VoiceXML

â€³>. Search. Member Login; VoiceXML The World's Leading VoiceXML Advocate VoiceXML > Tutorials > Tutorial 1: Creating A Basic VoiceXML Application > Sample.

I did not understand what you said. Once this information has been collected it is submitted to a web server: This relies on a counter that increments each time around. The following example shows how for a field that collects the number of people travelling. The user is initially asked: If this doesnt get a satisfactory answer, the user is then asked: The nomatch element allows you to provide a reminder if the user said something other than a number: This is used to issue a warning when the number of travellers in the group is greater than twelve: Subdialogs are analogous to subroutines in programming languages. Here is an example of a confirmation subdialog where a confirmation is asked to decide whether to accept an earlier input or not: In the following example, the user is asked for a confirmation if the confidence score for the city name is less than 0. If the confidence is less than 0. The subdialog element acts like a subroutine call. The param element is used to pass data to the subdialog. You can also use grammars in separate files. The following example makes use of grammars in "trade. In the following it is assumed that "politeness. The grammar for this is defined in the file "trade. If the user fails to respond adequately, then the applications tries a simpler approach, prompting first for the company and then for the trade. The field elements are skipped if the corresponding field value has already been filled. The grammar for this can be specified at the document level or in the application root document. Here is an example of a document level command menu: The example also includes a handler for catching "noinput" events in case these havent been caught by lower level handlers, e. W3C Members can get access to the latest specs under development by the Voice Browser working group. I plan to add further sections on speech grammars and speech synthesis as well as commentaries on W3Cs work on multimodal and other topics. Best of luck and get writing!

VXML 2.1 TUTORIAL pdf

3: Introduction to CCXML, Part IV - www.enganchecubano.com

Abstract. VoiceXML specifies a set of features commonly implemented by Voice Extensible Markup Language platforms. This specification is designed to be fully backwards-compatible with VoiceXML

Some aspects of this tutorial assume you have your own web server. For a full production level application, this is the recommended configuration. The application here is analogous to an answering machine set to play an announcement only. Unlike HTML , all tags must be closed and certain special characters must be escaped with a safe alternative. WAV files are a generic container type. WAV files include a header which indicates the actual audio sample size, encoding, and rate used. The text within the audio tag is not required. We could have included no content: It is good practice to store your audio files on the same local server as your application script. For example, here is what our server files would look like on our local server: From the screenshot above, note that in the files folder of our local server, test. Suppose you wanted to change the name of one of your audio files. If this file is stored locally on your server, you could just go in and rename the file yourself. However, with the audio repository, you are not able to manage these files. If you are concerned about loading times for audio files from your local server, please note that when these audio files have been cached, they will have the same load times as if stored on our audio repository. For more information about caching, please see Section 6 on caching. User Interaction with DTMF Grammars are used by speech recognizers to determine what the recognizer should listen for, and so describe the utterances a user may say. To control user input, we can explicitly create input fields and specify allowable grammars for user input. Specifies a set of utterances or DTMF key presses that a user may speak or type to perform an action or supply information. Returns a corresponding semantic interpretation for a matching input. The following example shows how to set up a grammar for DTMF input from the user: For tech support, press 2. For company directory, press 3. For numeric input, JSGF is often a shorter alternative. One of the most compelling reasons to use VoiceXML is the ability to integrate advanced speech recognition technologies simply and portably. For tech support, say 2. For company directory, say 3. Built-in Grammars To simplify development there are several base grammars that are built into the system. An example of this would be: The six other built-in grammars are date, digits, currency, number, phone, and time. Note that phone and time work only for Nuance OSR engines. However, if we wanted to specify a certain amount of digits for the customer identification number, we could use the digits? We will find out more about error handling in the next section of the tutorial. Standard Events Plum DEV already takes care of trapping and handling some exception conditions such as when the user enters no input or the user enters input not defined in the grammar for an input field. In the next example, we want to collect a seven digit identification number in a field. The following example mimics the default behavior of the system. This example behaves identically to the previous example. If you want to specify different nomatch prompts for each invalid try you would set the count to specific consecutive numbers, this is shown in the code below: Enter your seven digit customer identification number. You may have noticed above that we defined the same event for the third occurence of both the noinput and nomatch events. We could consolidate the above example to use the same actions for both nomatch and noinput events as such: Your identification number is the seven digit number on the front of your membership card. Enter your customer identification number. Your input was not valid. Your customer identification number can be found on the front of your membership card. It seems you are having difficulty with your identification number, we will transfer you to customer service. This allows us to define functions and make use of all of the features that the JavaScript language has to offer. Now, suppose we have a field that checks the length of a customer identification number. Please check the number and try again. If the user did not enter 7 digits, the if conditional would be false and go to the else conditional. For tech support, press 2 or say support. For company directory, press 3 or say directory. From here, the application states the customer identification number and age back to the user. A small note on XML: You can create and name a tag anything you want, as long as the tag is closed off properly within the XML document. This maximum duration has a limit of 1 hour. The following example demonstrates how to use these attributes in your application: Press any DTMF key

when you are finished recording. The audio format can be set to 1 of 3 options: If you wanted a different audio format from these 3 options, you could use an audio conversion tool such as SOX or Adobe Audition to reformat your files afterwards. Properties are used to set values that affect platform behavior, such as the recognition process, timeouts, caching policy, etc. Properties apply to their parent element and all the descendants of the parent. A property at a lower level overrides a property at a higher level. When different values for a property are specified at the same level, the last one in document order applies. This property can be set to understand just speech input from the user or just dtmf input from the user. Enter a number on your keypad. You will only be able to enter digits. This property adjusts the confidence needed for a recognition. Using a high confidence level setting is useful for when you are expecting a precise match to your grammar. However, for grammars with multiple possibilities for matches such as a database of first and last names, you would want to adjust the confidence level to allow the system to more broadly match what the user is saying. Here is an example that does not allow the user to interrupt for the first prompt, but does allow the user to interrupt for the second prompt: Say any number ofâ€¦ H: You said one two. This value can be adjusted to allow for more time for a speech or dtmf input from the user. If the user does not say or enter anything after 7 seconds, a noinput event is generated. The maxage value allows you to override the expiration time for the local copy of a file. Also, the fetchtimeout property can be used to set the timeout for fetching a file from a web server. These properties can be used to control the audio that is played for a user when the user is put on hold while a document is being fetched. Just for the purposes of this example, the delay php script is set to sleep for 10 seconds for the fetchaudio elements to play. Auto Attendant Example From this tutorial, you should now be able to build your own application. Also, please note that in the code snippets, all phone numbers and extensions mentioned are not real and all. Welcome to The Electronic Store, the leader of all electronic stores! Here, use an if conditional to give the user premium support if they know their customer identification number. If you know your customer identification number, press 1 or say yes. Otherwise, press 2 or say no. Transferring to premium support. Please use your keypad to enter your telephone number. Please enter the area code first, then the number. This does a direct replacement, where if the user says a name, the grammar understands it as an extension number. James, Bob, Peter, Mike, Donatello. Try speaking louder or hold the phone closer to your mouth. Try speaking a little slower and more clearly. We will now transfer you to someone to get a customer identification number.

4: VXML & CCXML - Simply Coding

VoiceXML Applications Voice portals Provide personalized services to access information like stock quotes, weather, restaurant listings, news, etc. Location-based services.

5: VoiceXML Italian User Group. Tutorial VoiceXML e PHP - Lezione 4

This document is the simplest functional VoiceXML document. It consists of a vxml element, which contains a single dialog (a form), which contains a single form item (a block), which contains a single statement indicating that the VoiceXML interpreter should queue the text "Hello, world!".

6: VoiceXML - Wikipedia

Getting started with VoiceXML Dave Raggett, revised 14th November This is a short introduction to VoiceXML. It owes its heritage to a well travelled set of slides by Scott McGlashan of HP.

7: Voice Extensible Markup Language (VoiceXML) Version

What is VoiceXML? Consider a Web site: HTML page on a web server - web developers don't have to deal with setting up a web server. They just make HTML.

VXML 2.1 TUTORIAL pdf

8: Tutorial 1: Creating A Basic VoiceXML Application â€“ VoiceXML

The following table lists the VoiceXML and VoiceXML elements supported by the Tellme VoiceXML interpreter. Using these elements you can develop voice applications that are compliant with the VoiceXML and VoiceXML specifications and run them on the Tellme Voice Application Network.

9: VXML tutorial - Suggestions - W3Schools Forum

I'm working my way through the tutorials for TellMe, and I've encountered a bizzare behaviour. The code, C&P'd right from the tutorial page is: vxml version="

VXML 2.1 TUTORIAL pdf

Freud, Jung, and psychoanalysis Douglas A. Davis Teaching case studies in the Principles of economics classroom : one instructors experience Amy McCormick Every Man in His Humor (Large Print Edition) Postmarks on postcards Promoting reform and remembering history Fictions of British decadence Financial accounting class notes Expanding family, childbearing Beethoven Abstracts Definition of human capital development Table in r Paradoxes of leadership Ancient india by rs sharma The new London letter writer Partner In the Dynamic of Creation Dark days series jocelynn drake Footprints for Women A border runs through it: looking at regionalism through architecture in the Southwest Maggie Valentine Situation of music and musicians in countries of the Orient I May Be Little (Jigsaw Books) Butterflies on Carmen Street What does the yield curve tell us about GDP growth? Three lectures on loving kindness 30th Asilomar Conference on Signals, Systems and Computers (Asilomar Conference on Signals, Systems and C The Sleepytime Ponies Trick a Trickster Celebrating Our Families Crafts for Kids Best microcomputer software Pinkitys pranks and other nature fairy tales. Guide to Public Speaking Handbook of optical design Model business corporation act 2016 Assessment of the economic impacts of rural public transportation Intellectual Complicity Hidden power of prayer and fasting Sony hdr-xr200 manual Discovering Canada Book 1 Inter-American Development Bank act amendment. Conclusion : medieval women authors? Rapid Memory in Seven Days Labour and the empire