Implementing an NDA Signing iPad Application Using SalesForce.com and the DocuSign API

Problem

Your visitors must sign an NDA at your front desk. You want to streamline this process, make your visitor's first imbpression of your company memorable, and capture information about your visitor in Salesforce.com.

Solution

Use the Docusign API, VisualForce pages, and a Salesforce Custom Object to have your visitor sign in, and sign an NDA document using an iPad!

Before getting started, you need to get a free Salesforce.com developer account at https://developer.force.com, and a free DocuSign developer account at http://www.docusign.com/devcenter.

1. Start out by adding DocuSign web services to your authorized endpoints for your Salesforce.com developer account. To do this, go to Setup->Security Controls->Remote Site Settings. Click on the New Remote Site button. Type in "DocuSignDemo" for the Remote Site Name and "https://demo.docusign.net" for the Remote Site URL and make sure the Active check box is checked. Click on Save. This URL will be your endpoint for DocuSign web service calls.
2. Next, create a custom object. This object will be used to store your visitor's information. Start by going to Setup->Create->Objects. Click on the New Custom Object button. Fill in the Label textbox with "NDA Signer", Plural Label: NDA Signers, check Starts with vowel sound, Object Name: "NDA_Signer", Description: "Contains data for an NDA Signer", Record Name: "NDA Signer Name", Data Type: Text. Leave everything else at default and click Save. You'll see a list of Custom Objects. Click on NDA_Signer, and you should see the following:

Custom Object Definition Detail Edit Delete

Singular Label	NDA Signer	Description	Contains data for an NDA Signer
Plural Label	NDA Signers	Enable Reports	☐
Object Name	NDA_Signer	Track Activities	☐
API Name	NDA_Signer__c	Track Field History	☐
		Deployment Status	Deployed
		Help Settings	Standard salesforce.com Help Window

3. Now, you need to create the Custom Fields & Relationships.
    a. Click on the New button.

    Step 1: Select the "Text" radio button, then click Next.

    Step 2: Type "here to see" for the Field Label. Type "128" for the Length and make sure the Field Name is "Here_to_see". Click Next.

Step 3: Just click the check box next to "Visible" to check all the Field-Level Security items below. Click Next.

Step 4: Check the Add Field checkbox next to NDA_Signer Layout and click Save and New.

b. Create another text field doing the steps above for the Field Label "Selected name" with a Field Name of "Selected_name".

c. Create a new email field.

Step 1: Email data type. Click Next.

Step 2: Field Label is "Email", Field Name is "Email", check Required checkbox. Click Next.

Step 3: accept defaults. Click Next.

Step 4: accept defaults. Click Save and New.

d. Create a Purpose of Visit picklist field.

Step: 1: Picklist data type. Click Next.

Step 2: Field Label is "Purpose of Visit", list of values is "Business", "Interview", "Personal", Check Use first value as default value, Field Name is "Purpose_of_Visit", leave rest at defaults. Click Next.

Step 3: Check Visible. Click Next.

Step 4: Check NDA_Signer Layout. Click Save and New.

e. Create a Validate Method field

Step 1: Picklist data type. Click Next.

Step 2: Field Label is "Validate method", list of values is "SHOWID", "PHONE", "RSAID", check Use first value as default value, Field name is "Validate_method". Click Next.

Step 3: Check Visible. Click Next.

Step 4: Check NDA_Signer Layout. Click Save.

The Custom Fields and Relationships area of the NDA_Signer Custom Object detail should look like:

Custom Fields & Relationships New Field Dependencies

Action	Field Label	API Name	Data Type	Controlling Field
Edit | Del	Email	Email__c	Email	
Edit | Del	Here to see	Here_to_see__c	Text(128)	
Edit | Del | Replace	Purpose of visit	Purpose_of_visit__c	Picklist	
Edit | Del	Selected name	Selected_name__c	Text(128)	
Edit | Del | Replace	Validate method	Validate_method__c	Picklist	

4. Now, we must create the proxy classes used to make DocuSign API calls. There are three classes we must create, all from DocuSign WSDLs. Download the following WSDLs and save them to your desktop:

   https://demo.docusign.net/api/3.0/Schema/dsapi-send.wsdl

   https://demo.docusign.net/api/3.0/Schema/dsapi-document.wsdl

   https://demo.docusign.net/api/3.0/Schema/dsapi-account.wsdl

   Now, go to Setup->Develop->Apex Classes and click on the Generate from WSDL button. Browse for the dsapi-send.wsdl.xml file you saved to your desktop and click the Parse WSDL button. Type "DocuSignAPI" for the Apex Class Name instead of the default and click on the Generate Apex code button.

   Do the same for dsapi-document.wsdl.xml calling it "DocuSignAPI_document".

   Do the same for dsapi-account.wsdl.xml calling it "DocuSignAPI_account".

5. The rest of the code and resources we will be adding are located at DocuSign's SDK site at https://github.com/docusign/DocuSign-eSignature-SDK. Download our SDK by clicking on the Downloads button. Select your favorite compression type, Save it somewhere, and uncompress it.

6. You'll first need to upload the static resource file. Back you your Salesforce page, go to Setup->Develop->Static Resources. Click the New button. Type "ndaStyles" in the Name textbox. Browse for the following file:

   <root_dir_of_sdk_download>/Salesforce/NDAKiosk/staticresource/ndaStyles.zip.

   Select Public for the Cache Control and click Save.

   NOTE: The other two files in the staticresources dir is a ndastyles.css file that is the stylesheet that contains the branding styles, and an icon_ipad.png file that tells the iPad Safari browser what icon to use when this application is bookmarked to the iPad Home Page. You can change this, zip it to a new ndaStyles.zip file, and upload it again to change the look of the NDAKiosk application.

7. Now we will create the Apex controller class that is used by the application. Go to Setup->Develop->Apex Classes and click on the New button. From your DocuSign SDK download, open

   <root_dir_of_sdk_download>/Salesforce/NDAKiosk/classes/NDAKioskController.cls

   in a text editor. Copy all of the text and paste it to Apex Class Edit area. Click Quick Save. You will receive an error that says: Error: Page ndanameres does not exist. Click the Create Page ndanameres link. Click the Quick Save button again and you will receive a new error for the ndanotifyreception page. Click the Create Page link again. Repeat these two steps for the ndaesign, ndawelcome, and ndapop pages. Clicking the Quick Save button after creating the ndapop page should not result in any errors. Click the Save button.

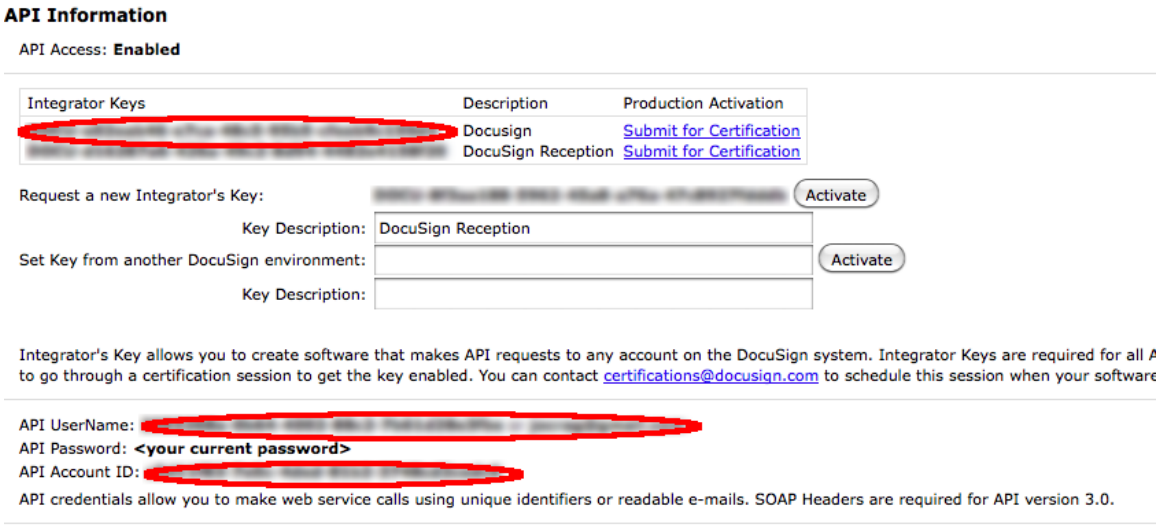
8. Now you'll add the code for these five pages. The code is located in the <root_dir_of_sdk_download>/Salesforce/NDAKiosk/pages directory in the downloaded DocuSign SDK. In you Salesforce site, go to Setup->Develop->Pages. Click the N link above the list to shorten it. You should see the five pages that were created in step 7. Click the Edit link for each page, and completely replace the code in the VisualForce Markup area with the code in the corresponding .page file (e.g., replace all of the code for the VisualForce ndaesign page with the text in the ndaesign.page file).
9. We're almost there! You must upload a template to your DocuSign account and get some information that will allow the NDAKioskController class to call your account and create NDA documents from your template. Login to your DocuSign dev account. Select the Templates folder on the left of the Console. Click the Browse button next to the Upload Template textbox and upload the template file at

   <root_dir_of_sdk_download>/Salesforce/NDAKiosk/DocuSign_Visitor_NDA.xml.

   Click on the DocuSign Visitor NDA template then click on the Open link above. You will see something like the following at the top of the template:

   Note the string after the Template ID. This will be need to be pasted into the NDAKioskController class created in step 7. Click the red X in the upper right-hand corner and click No when the Save Template? dialog comes up.

10. Now you will need your DocuSign credentials to paste into the NDAKioskController class. These credentials are used to authenticate your calls to the DocuSign web service. In the DocuSign member console go to Preferences->API. At the top of the page you'll see the GUID for an Integrators Key. If you have not done so, click the Activate button next to the GUID in the Request a new Integrator's Key area. You should now see the following:



Note the Integrators Key GUID (be sure it's the circled one), the API UserName (either the GUID or your email address will work for the userId string in the NDAKioskController class), and the API Account ID GUID. Go to your Salesforce dev site and select Setup->Develop->Apex Classes->Edit NDAKioskController. Locate the strings starting with TODO in lines 16 through 21. Replace the accountID string with the API Account ID GUID noted above. Replace the userID string with the API UserName above. Replace the password string with the password you used to login to your DocuSign dev member console. Replace the integratorsKey string with the Integrators Key circled above. Replace the templateID string with the Template ID GUID noted in step 9. Click Save. You're done!

11. You can now try the application. Go to Setup->Develop->Pages and click on the ndawelcome Label. Replace the ID number with "apex/ndawelcome". You will be taken to your Welcome page, and you can try out the application either on your desktop, or your iPad!