Entwicklercamp 2011

Lotus knows.

Smarter software for a Smarter Planet.

Track 1 Session 5

XPages mit Desktop-Features ausstatten

Karsten Lehmann | Geschäftsführer | Mindoo GmbH

NOTES & DOMINO ENTWICKLERCAMP

Agenda

- Einführung
- Die Umgebung kennen
- Desktop-Features nachbilden
- Echte Desktop-Features nutzen
- Zusammenfassung
- Q&A

Über uns

- Mindoo ist IBM Business Partner und Notes/Domino Design Partner
- Wir konzentrieren uns auf die „neuen“ Aspekte der IBM Lotus Notes-Entwicklung
  - Eclipse/Expeditor-Plugins und Rich-Client-Anwendungen
  - Composite Application Architekturen
  - LiveText Erweiterungen
  - XPages-Anwendungen
- Karsten Lehmann und Tammo Riedinger
  - Gründer der Mindoo GmbH
  - Seit 2004 Entwickler der Applikation MindPlan® für die Haus Weilgut GmbH, Mindmapping und Projekt-Management für Lotus Notes, IBM Award Winner 2008
- Weitere Informationen:
- http://www.mindoo.de

[mindu:]
Mindoo GmbH

Die Cloud – das Ende für den Desktop?

- Web-Technologien modern und plattformübergreifend
- Trend zu “Cloud-based Services”
  - Dienste von überall per Web zugänglich
  - Zentralisiert auf gehosteter Hardware
  - Arbeit vollständig im Browser erledigen
  - Zugang mit Notebook und mobilen Endgeräten (Smartphone, Tablets etc.)
- XPages im Domino-Umfeld mächtiges Werkzeug, um Anwendungen webfähig zu machen
- Entwicklung mit XPages liegt im Trend
  - Allein 11 Sessions beim Entwicklercamp 2011 zum Thema XPages!

Die Herausforderungen

- Anwendungen in der Cloud scheinen den lokalen Desktop zu verdrängen
- In der Realität leider noch mit Kinderkrankheiten, Desktop hat weiterhin seine Daseinsberechtigung
- Fehlende Offlinefähigkeit der Cloud-Lösung
  - Nicht jeder ist immer online
  - Was ist, wenn der Server down ist?
- Ressourcenmangel in der Cloud
  - Nicht alle anzubindenden Systeme haben eine Web-Schnittstelle oder sind vom Web erreichbar (z.B. Warenwirtschaftssysteme, SAP Client)
  - Systeme und Dateien z.T. nur lokal verfügbar aus Sicherheitsgründen
  - Bestehende Codebasis nutzt andere Technologien zur Kommunikation mit anderen Anwendungen: COM, Java-Schnittstellen, Kommandozeilenübergabe von Daten

Die Herausforderungen

- Performancemangel in der Cloud
  - Laufzeit für aufwendige Operationen zu lang (10.000 Dokumente aktualisieren ohne Fortschritt, Session läuft ab)
  - Last auf Server bei paralleler Nutzung durch viele User zu hoch
- Usability ausbaufähig
  - Nutzer sind mehr gewöhnt als Web-Anwendungen
  - HTML5 bietet gute Ansätze für mehr Benutzerfreundlichkeit:
    - Ajax für dynamische Weboberflächen
    - Manifestdatei/lokale DB für Caching und Offlinefähigkeit von Daten
    - File API für Drag&Drop von Dateien in den Browser
  - Lokale Anwendungen wegen jahrelanger Koexistenz besser integriert
    - Übergabe von Dateien per Copy&Paste
    - Gemeinsam genutzte Festplatte und Dateiformate
    - Schnittstellen zwischen den Anwendungen
- Fehlende Standards für Web-Anwendungen

Ein Lösungsvorschlag

- Hybridsysteme: Dieselbe Anwendung lokal und im Web nutzen
- Mit XPages im Client (XPiNC) endlich möglich!
- Bei lokaler Ausführung Nutzung von Desktop-Features
- Usability im Web schrittweise erhöhen, sobald neue Technologien verfügbar sind

- Diese Session beschreibt Wege, Desktop-Features im Web nachzubilden und echte Desktop-Features in lokalen XPages-Anwendungen zu nutzen

Agenda

- Einführung
- Die Umgebung kennen
- Desktop-Features nachbilden
- Echte Desktop-Features nutzen
- Zusammenfassung
- Q&A

Die Umgebung kennen

- XPages-Elemente abhängig vom Browser und der Plattform anzeigen/ausblenden
- Rendered-Attribut lädt ein Control, zeigt es jedoch nicht an
- Loaded-Attribut verhindert bereits das Laden eines Controls
  - Sinnvoll z.B. Bei Unverträglichkeit von Code mit bestimmten Notes-Versionen
- Prüfung auf Web/Notes-Client mit @ClientType
  - @ClientType==”Notes” → XPage im Notes-Client
  - @ClientType==”Web” → XPage im Web

```
<xp:button value="Login" id="loginButton"
rendered="#{javascript:@ClientType=='Web'}"></xp:button>
```

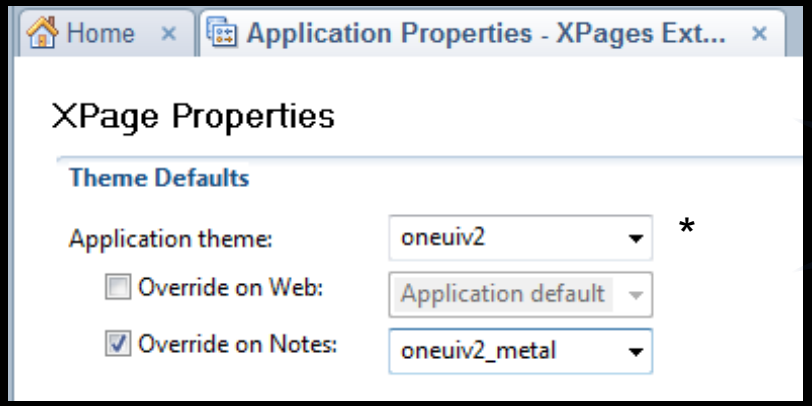

Die Umgebung kennen

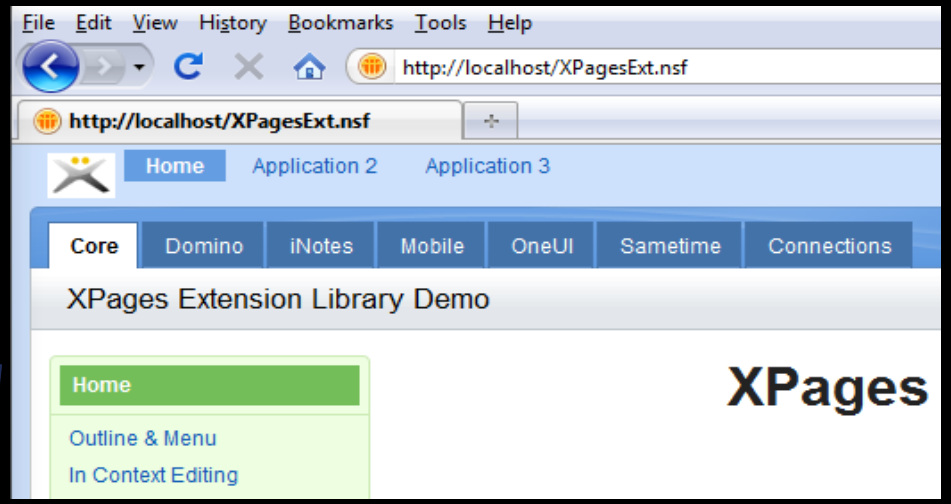
- Prüfung auf Betriebssystem des XPages-Servers:
  - Java-Property os.name auswerten: `System.getProperty("os.name")`
- Prüfung auf bestimmte Endgeräte durch Auswerten des User-Agent-Strings
  - `context.getUserAgent().isIE() / isFirefox()` etc.
  - Oder manuelle Prüfung von `context.getUserAgent().getUserAgent()`:

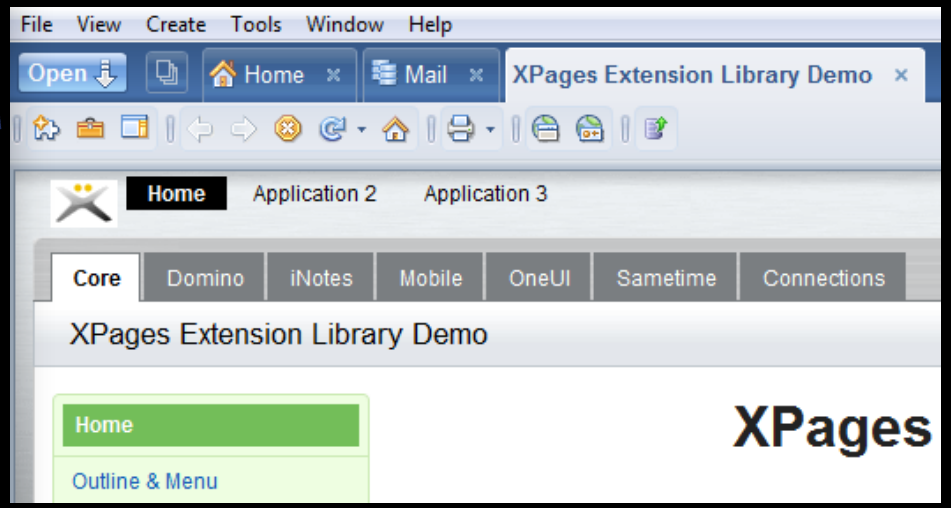
```
<!-- Dynamische Ressource in XPages-Theme:
     iPhone.css nur für iPhones einbinden -->
<resource
rendered="#{javascript:context.getUserAgent().getUserAgent().ma
tch('iPhone')}">
    <content-type>text/css</content-type>
    <href>iPhone.css</href>
</resource>
```


Die Umgebung kennen

- Verschiedene Themes für Web/Client verwenden:
- Optische Anpassung der XPages-UI an den Notes-Client möglich



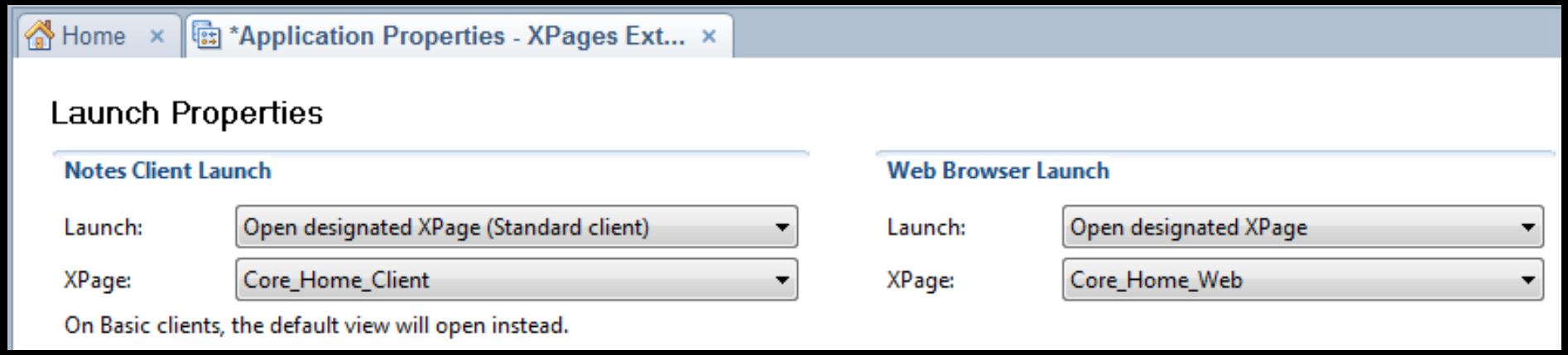




* Im Standard verfügbare OneUI-Varianten in 8.5.2: oneuiv2, oneuiv2_green, oneuiv2_gold, oneuiv2_metal und oneuiv2_red

Die Umgebung kennen

- Verschiedene XPages öffnen beim Starten derselben Datenbank in Web/Notes:



Agenda

- Einführung
- Die Umgebung kennen
- Desktop-Features nachbilden
- Echte Desktop-Features nutzen
- Zusammenfassung
- Q&A

Desktop-Features nachbilden

- Erhöhen der Benutzerakzeptanz durch eine ansprechende, Desktop-nahe Bedienung der Anwendung
- Lösungen funktionieren lokal und auch im Web

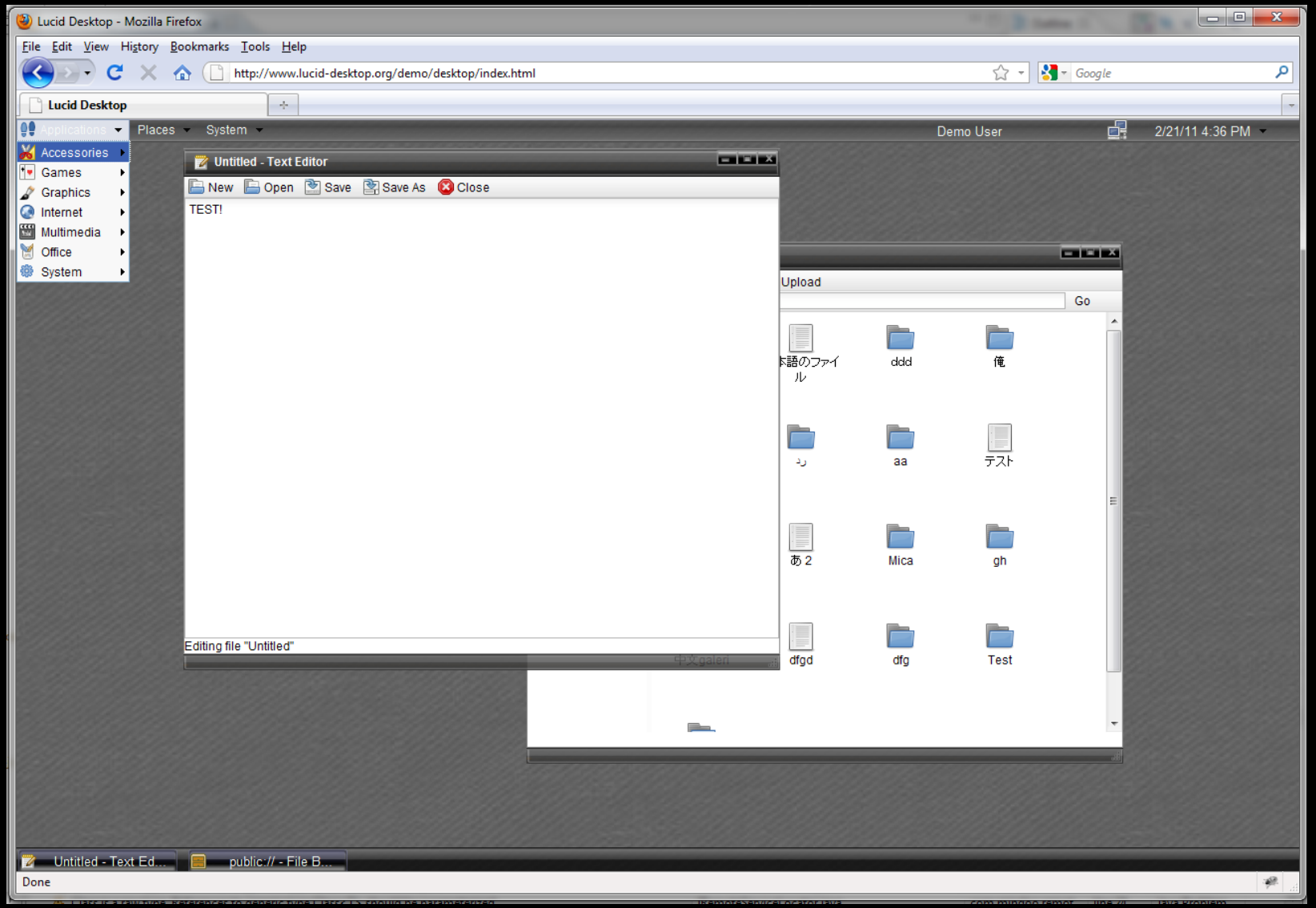
Web Desktops

- Windows-ähnlicher Desktop mit Fenstern für das Web
- Nützlich bei der Kombination mehrerer Web-Anwendungen in derselben Oberfläche
- XPages-Anwendungen liefern Fensterinhalte und profitieren von gemeinsamer Infrastruktur
- Viele kommerzielle und viele freie Implementierungen verfügbar:
  - http://de.wikipedia.org/wiki/Web_Desktop
- Frei
  - Lucid Desktop – PHP-basiert, verwendet Dojo
  - JavaScript Window Manager – verwendet Scriptaculous Library
  - Prototype Window Class
- Kommerziell
  - ExtTop – Beispiel-Implementierung auf Basis von Ext.JS
  - CEITON WinLIKE – Web Window Manager

Demo

- Web Desktop

Web Desktop - Screenshots



Dateiformate generieren

- Eine Vielzahl von Dateiformaten lässt sich mit Hilfe von Bibliotheken lesen und schreiben
- Funktionsumfang ggf. eingeschränkt, Dateigenerierung im Backend ohne UI und Anpassungsmöglichkeit
- Office-Dokumente produzieren
  - Microsoft Office: Apache POI Projekt (frei)
  - OpenOffice/Symphony: ODF Toolkit (frei)
- PDF-Dateien generieren
  - Apache FOP: XML-PDF-Konvertierung (frei)
  - iText PDF: API zur PDF-Generierung (frei/kommerziell)
- MS Project
  - MPXJ-Bibliothek: Datenexport/-manipulation (frei)

Zugriff auf lokalen Rechner

- Lesen und Schreiben von Daten erfordert mehr Rechte, als der Browser einer Webanwendung einräumt
- Workaround über Browser-Plugins: ActiveX (IE), signierte Applets (Java-Plugin), Flash (große Verbreitung)
- JavaScript-Kommunikation zwischen Weboberfläche und Browser-Plugin z.B. über LiveConnect-Schnittstelle
- Moderner Ansatz: HTML5, z.B. File-Upload per Drag&Drop mit „File API“
  - Browser-Support noch sehr eingeschränkt (Firefox 3.6, Chrome 7, Safari 6)

Agenda

- Einführung
- Die Umgebung kennen
- Desktop-Features nachbilden
- Echte Desktop-Features nutzen
- Zusammenfassung
- Q&A

Echte Desktop-Features nutzen

- Auf der Plattform verfügbare Dienste und Programme nutzen, um Mehrwert zu generieren
- Zugriff auf lokale Umgebung und APIs erfordert z.T. einige Tricks:
- XPages-Code im Client läuft mit eingeschränkten Berechtigungen
  - SecurityManager verhindert bestimmte Java-API-Aufrufe
  - Einige Operationen (z.B. Plattenzugriff) führen lediglich zu ECL-Hinweisdialogen
  - Andere werden direkt mit einer SecurityException unterbunden:

```
java.lang.SecurityException:
    ECL Permission Denied (java.awt.AWTPermission
    showWindowWithoutWarningBanner)
com.ibm.domino.xsp.module.nsf.platform.NotesPlatform.checkPermission(
    NotesPlatform.java:91)
COM.ibm.JEmpower.applet.XPagesSecurityManager.checkPermission(
    XPagesSecurityManager.java:131)
COM.ibm.JEmpower.applet.XPagesSecurityManager.checkPermission(
    XPagesSecurityManager.java:107)
    java.awt.Desktop.checkAWTPermission(Desktop.java:233)
    java.awt.Desktop.open(Desktop.java:261)
```


XPiNC-Berechtigungen unter Notes 8.5.1

- Viele Java-API-Aufrufe führen zu SecurityExceptions
- u.a. Parsen von XML und Netzwerkkommunikation (z.B. Web Services, nur Socket-Verbindungen möglich)
  - In beiden Fällen fehlt die „NetPermission“ zum Zugriff auf URLs und das Netzwerk
- Java-Workaround für beschränkten Zugriff:
  - JAR mit Code nach lib/ext kopieren
  - Im Code `AccessController.doPrivileged()` nutzen, um Code mit mehr Rechten auszuführen
  - Bibliotheken in lib/ext werden als JVM-Erweiterung geladen und dürfen daher die Sicherheitsprüfung des SecurityManagers umgehen
  - Automatische Verteilung der JAR-Datei problematisch
- Alternative: Umweg über LotusScript
  - Öffnen einer Notes-Page aus der XPages-Anwendung:

```
document.location.href=
    "notes://server/dbpath/Pagename?OpenPage"
```

  - LotusScript-Code greift auf System zu oder ruft per LS2J Java-Code auf

XPiNC-Berechtigungen

- Aufrufsyntax zum Ausführen von Code mit mehr Rechten:

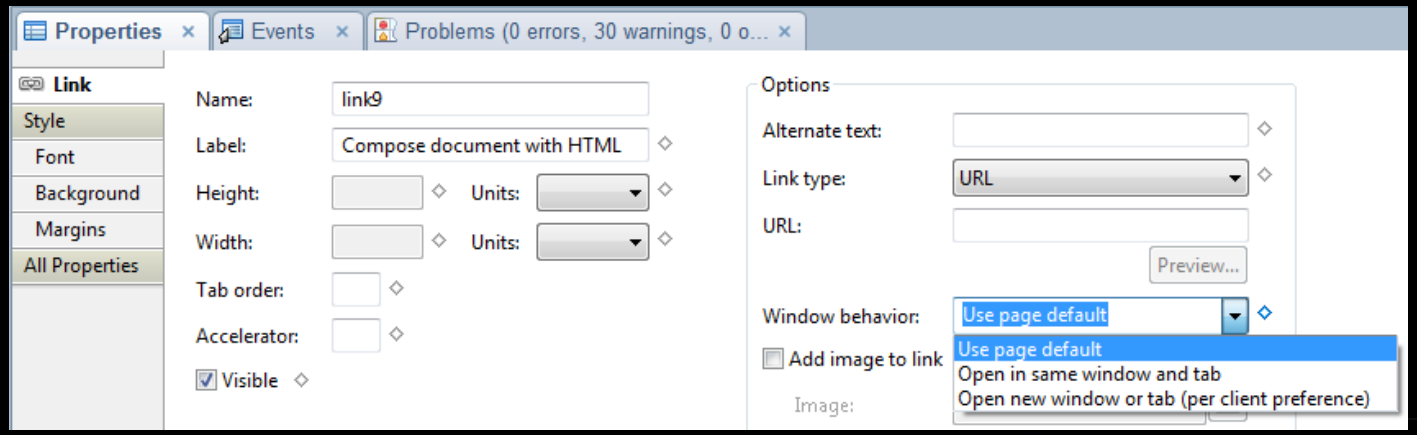
```
T result=
 AccessController.doPrivileged(new PrivilegedAction<T>() {
  public T run() {
    // Code läuft ohne Prüfung des SecurityManagers.
    // Achtung: kein Sicherheitsleck öffnen!
   T newT=buildT();
   return newT;
  }
 });
```


XPiNC-Berechtigungen unter Notes 8.5.2

- Mehr Permissions im Standard (u.a. NetPermission)
- Ggf. noch Probleme mit Standard-Bibliotheken, die z.B. den ClassLoader nutzen/ändern
- XPages Extension API lässt sich als Workaround nutzen:
  - Neu in Lotus Notes 8.5.2
  - Erweiterung von XPages mit zusätzlichen Komponenten (UI und Backend)
  - Eclipse-Plugins auf Client und Server erforderlich
  - Plugins haben volle Rechte (durch AccessController-Aufruf) und können vom XPages-Code nahtlos verwendet werden (eine JVM)
  - Verteilung von Plugins wesentlich einfacher möglich (per Policy)
  - Weitere Details und Demo später

Einsatzbereiche im Notes Client

- XPages-Elemente können seit Notes 8.5.2 an mehreren Stellen im Client eingesetzt werden
- XPages-Komponenten in der Sidebar:
  - In 8.5.2 nur in CA möglich, wahlweise wenn die CA sichtbar ist oder wenn sie als Reiter geöffnet ist
  - Können per CA-Wire Daten empfangen und darauf reagieren
  - In 8.5.3 ist voraussichtlich das Verbinden von LiveText mit XPages-Sidebar-Komponenten möglich
  - z.B. Klick auf Helpdesknummer in Email zeigt Helpdeskfall als XPage in Sidebar
- Öffnen von XPages-Inhalten in mehreren Reitern
  - Target-Property für Links in 8.5.2



Demo

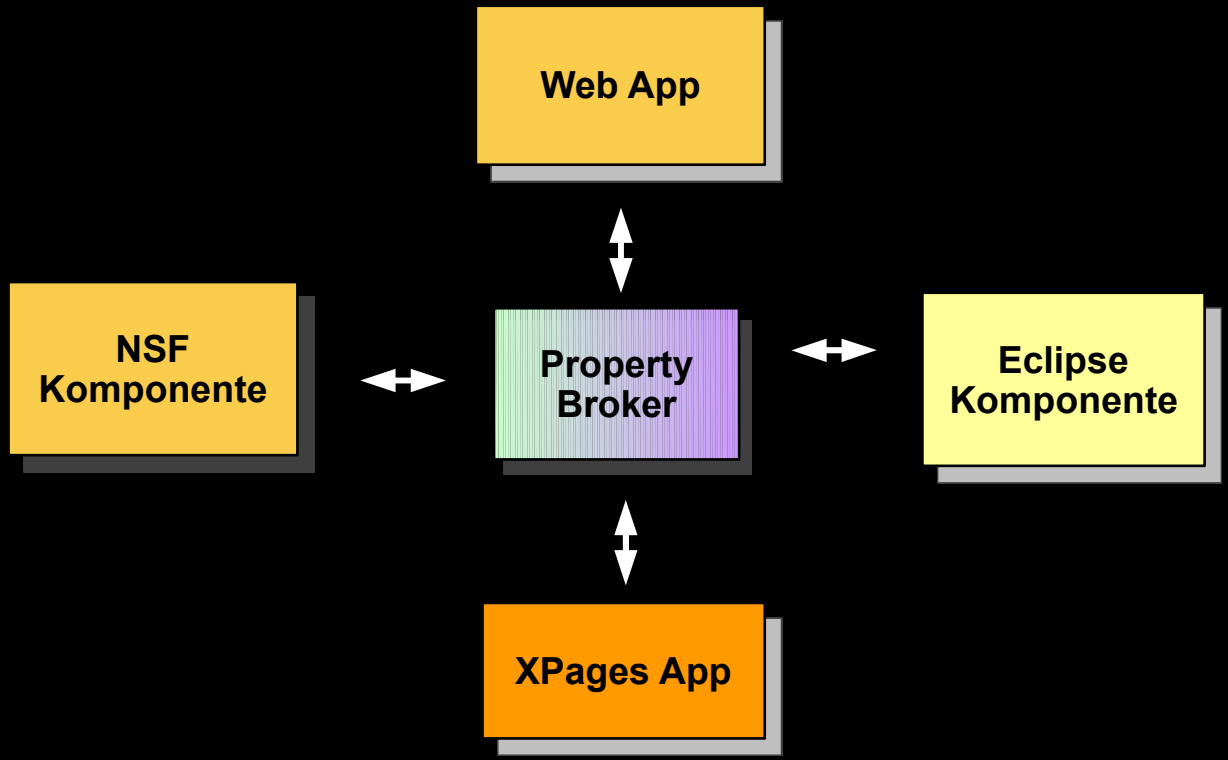
- XPages in Sidebar
- Target-Property für Links

Interaktion mit anderen Anwendungen

- Asynchrone und synchrone Interaktion zwischen XPages und anderen Anwendungen
- Beispiele für asynchrone Verbindungen:
  - Composite Application
- Synchrone Verbindungen:
  - Aufrufen von Anwendungen per Kommandozeile
  - COM-Schnittstelle unter Windows
  - APIs über Netzwerkprotokoll

Composite Application

- Lose Kopplung einer XPages-Anwendung mit anderen CA-Komponenten
- Datenübertragung erfolgt asynchron, String senden/empfangen mit Property Broker als Vermittler
- Integration von Windows-Anwendungen möglich mit kommerzieller Lösung OpenSpan oder Eigenentwicklung



Demo

- XPages in Composite Application
  - Erweiterung der Mail-CA
  - Kopplung von XPages-Forum mit Browser und Notes-Ansicht

Aufruf von Anwendungen

- XPages-Anwendung legt Datei lokal ab und startet sie in der verknüpften Anwendung
- Starten von Webseiten in Standardbrowser
- Plattform-Unterscheidung erforderlich, da Standardweg seit Java 1.6 (Java Desktop API) selbst in 8.5.2 noch zu SecurityException führt
- Ausführen von Anwendungen über Java: `Runtime.getRuntime().exec()`

COM-Schnittstelle unter Windows

- Verwendung von COM-Schnittstelle von LotusScript aus direkt möglich
- Für Java existieren diverse Java-COM-Brücken
- Erfordern eingebundene DLL, Verteilung und Berechtigung kniffelig
- Nutzbar mit Extension API-Plugin, da Plugins nativen Code mitbringen können und zum Laden berechtigt sind
- Alternative DCOM:
  - COM über Netzwerkprotokoll
  - Ohne DLL möglich, muss jedoch als Dienst unter Windows aktiviert werden
  - Java-DCOM-Brücke per TCP/IP: J-Interop

Demo

- Word-Export aus XPages über COM in LotusScript

APIs über Netzwerkprotokoll

- Beispiel: UNO API von OpenOffice und Lotus Symphony
  - Plattformübergreifend verfügbar
  - UNO API relativ kompliziert
    - „Nice Office Access“ wesentliche Vereinfachung, Wrapper für UNO API
- Beispiel: JDBC-Schnittstelle zu SQL-Datenbank

Zugriff von Eclipse-Plugins auf XPiNC

- Über Eclipse-APIs erhalten Plugins Zugriff auf den Bereich, der die XPage anzeigt (XspViewPart)
  - Abfragen von aktueller URL, Datenbankpfad und des XPage-Titels
  - Lese- und Schreibzugriff auf den DOM-Baum
  - Ausführen von JavaScript auf der aktuellen Seite
  - Seitennavigation kann per LocationListener detektiert und abgefangen werden
    - Damit Informationsrückgabe an Eclipse möglich
- Über Eclipse-Selektion teilt XspViewPart die Seitenwechsel mit
  - Name, URL und Titel
- Mögliche Einsatzszenarien:
  - Kontexthilfe in der Sidebar
  - Starten von Eclipse-Funktionen aus Client-seitigem JavaScript

Zugriff von Eclipse-Plugins auf XPiNC

```
IWorkbenchPart part=PlatformUI.getWorkbench().
   getActiveWorkbenchWindow().getPartService().getActivePart();

if (part instanceof XspViewPart) {
      XspViewPart xPart=(XspViewPart)part;
      XspXulRunnerBrowser browser=xPart.getWebBrowser();
      XspBrowserWrapper wrapper=xPart.getBrowserWrapper();

      //read information about URL and database name
      String notesUrl=wrapper.getCurrentNotesUrl();
      String dbTitle=wrapper.getDatabaseTitle();
      String dbPath=wrapper.getNSFName();

      //get read/write access on HTML DOM tree
      HTMLDocument doc=browser.getDocument();
}
```


Demo

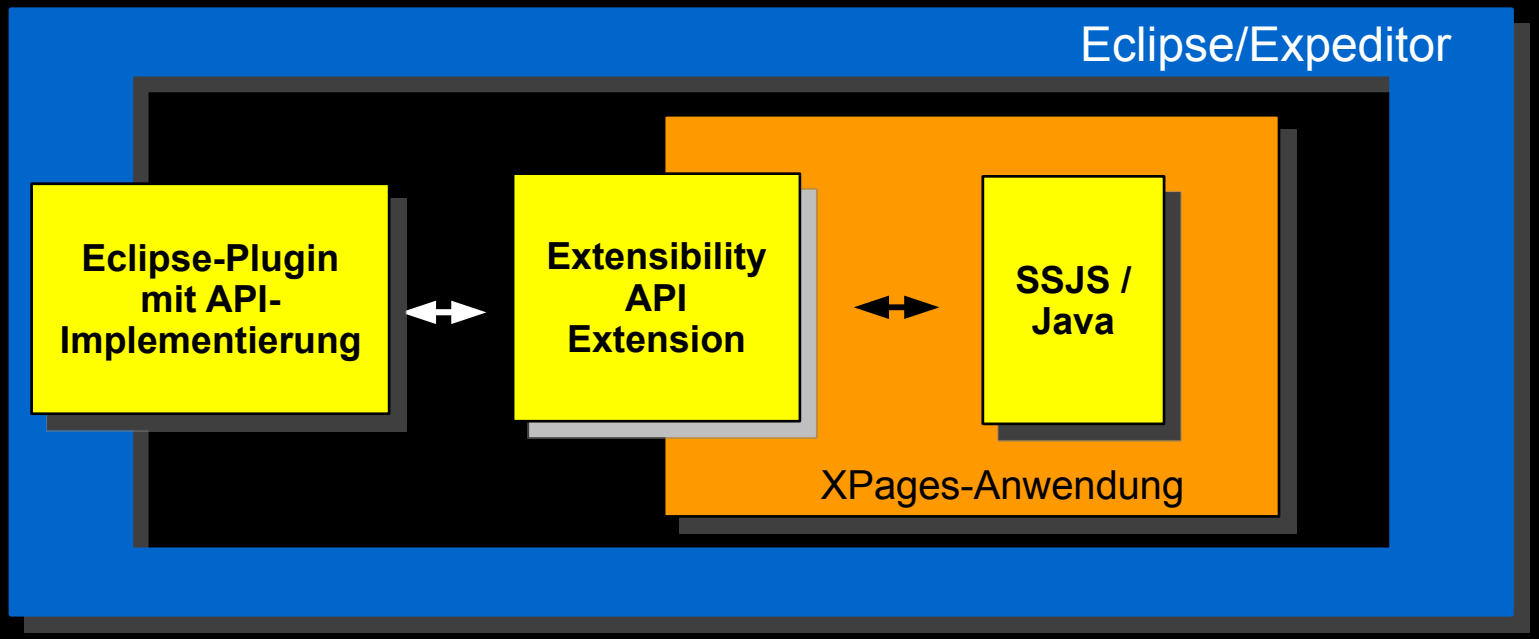
- Firebug zu XPiNC-Anwendung hinzufügen per Plugin
- Drag&Drop von Dateien in XPiNC-Fenster
- Kontexthilfe zu XPages-Anwendung

XPages Extension API

- Erweiterung von XPages-Runtime um zusätzliche Komponenten für UI und Backend
- Populäres Beispiel: XPages Extension Library von IBM (Extlib)
  - Vielzahl von zusätzlichen XPages-Controls, die dem Entwickler viel Arbeit und Zeit ersparen
  - u.a. fertig konfiguriertes OneUI-Layout, Dojo-Controls und iNotes-Kalender-Komponente

XPages2Eclipse Toolkit

- Verwendung der Extension API, um Client-spezifische Funktionen in XPages-Anwendungen aufzurufen
  - Plugin stellt Brücken-Control zur Verfügung, das im DDE in die XPages-Anwendung eingebunden wird
  - Aufgerufener Code im Control hat Zugriff auf Eclipse APIs
  - Gibt XPages-Anwendungen UI-Zugriff auf Eclipse, klassische Notes-Designelemente und Symphony



Demo

- XPages2Eclipse
  - UI-Zugriff auf Notes-Elemente
  - Jobs-API für Nebenläufigkeit
  - Registrieren von JavaScript-Code als Listener im Notes Client
  - Symphony-Export

Zusammenfassung

- Nutzung lokaler Dienste und Programme verbessert Benutzerakzeptanz und Funktionsvielfalt von XPages-Anwendungen
- Anwendung sollte möglichst in Web und Notes lauffähig sein mit erweiterter Funktionalität bei lokaler Nutzung
- XPages Extension API in Notes 8.5.2 vereinfacht die Nutzung von Desktop-Features wesentlich

Vielen Dank!

Zeit für Fragen