Beschrijving JabberPoint met inwerkopdracht

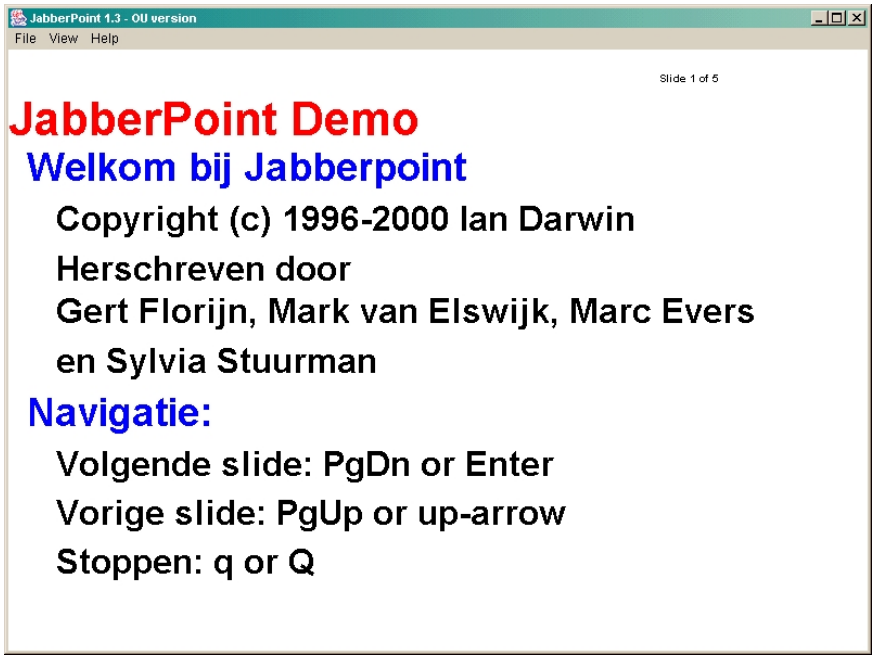
1 Algemeen overzicht

JabberPoint ondersteunt het tonen van presentaties die worden ingelezen vanuit een bestand (XML- of textformaat). Er kan dus niet "ge-edit" worden.
Om JabberPoint te starten moet de klasse JabberPoint worden geactiveerd met als optioneel argument de naam van een presentatiebestand. Jabberpoint heeft twee bibliotheken nodig: jdom.jar en xerces.jar. Mogelijke aanroepen vanaf de command-line zijn dus:

java -cp .:jdom.jar;xerces.jar JabberPoint test.xml of
java -cp jabberpoint.jar;jdom.jar;xerces.jar JabberPoint test.xml

Het weglaten van een argument zorgt ervoor dat een ingebouwde presentatie wordt vertoond.

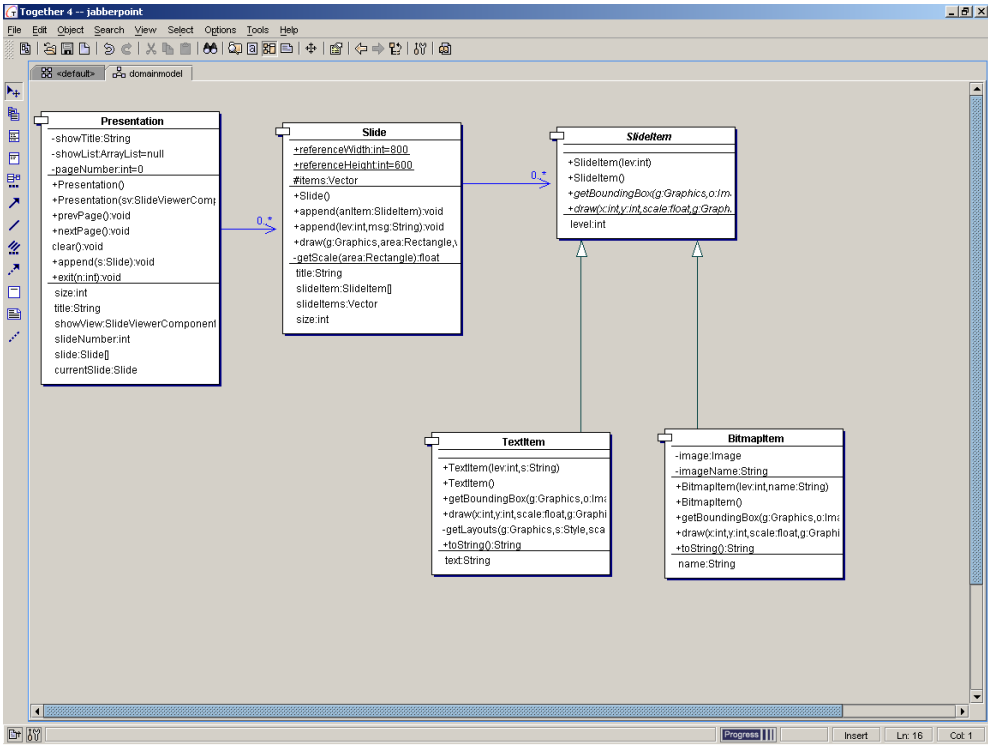
Na het opstarten van JabberPoint (hier onder zonder argumenten) verschijnt de eerste slide uit de geladen presentatie.



FIGUUR 1 Startscherm van JabberPoint

De gebruiker kan nu navigeren via de aangegeven toetsen of via het viewmenu. In het filemenu zijn enkele opties beschikbaar die automatisch ingebouwde acties uitvoeren.

2 De domeinklassen



FIGUUR 2 Domeinobjecten binnen JabberPoint

Figuur 2 laat de belangrijkste domeinklassen van JabberPoint zien. Kernbegrip is de *Presentation* die is opgebouwd uit *Slides*. Elke *Slide* representeert één pagina op het scherm. Een *Slide* heeft een titel (die bovenaan de slide wordt vertoond) en een aantal *SlideItems*, de elementen op de slide die onder elkaar worden vertoond. *SlideItem* is een abstracte klasse; concrete implementaties ervan zijn *TextItem* en *BitmapItem*. Een *TextItem* representeert een stukje tekst, een *BitmapItem* een plaatje (uit een file waarvan de naam in het *BitmapItem* object is opgenomen).

Ter illustratie toont figuur 3 een screenshot van JabberPoint in actie. (JabberPoint is hier opgestart met als argument de bestandsnaam "test.xml". Het programma toont hier een *Slide* met de titel "Achtergrond" (weergegeven in rode tekst). De *Slide* bevat de volgende sequentie van *SlideItems*:

– Een *TextItem* met de tekst "JabberPoint is een oefening ... universiteit"
– Een *BitmapItem* dat verwijst naar een image-bestand met het logo van de OU.
– Een *TextItem* met det tekst "De oefening ... SERC"
– Een *BitmapItem* dat verwijst naar een image-bestand met het logo van SERC.

FIGUUR 3 JabberPoint in actie met test.xml

Elk *SlideItem* op een slide is geassociëerd met een level. Dit level geeft het "nestingsniveau" van de items aan en wordt vooral gebruik bij het bepalen van de presentatievorm of *Style* van het item, dat wil zeggen: horizontale indentatie, font, kleur, en fontgrootte. De verschillende beschikbare stijlen worden gedefinieerd in de klasse *Style*.

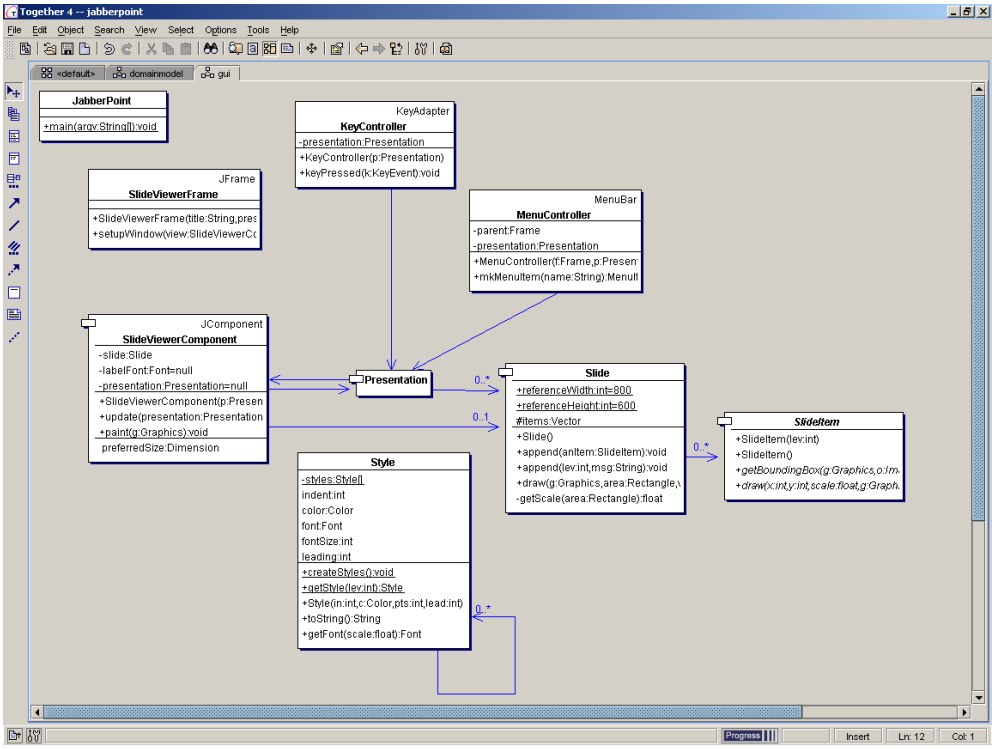
Ter illustratie: in figuur 3 staan de *TextItems* op level 1, terwijl de *BitmapItems* op level 2 zijn gedefinieerd. Level 0 wordt door het programma gebruikt voor de weergave van *Slide* titels (maar dat gebeurt pas bij het tekenen, zie inwerkopdracht).

Presentation en *Slide* bevatten basisoperaties om de domeinobjectenstructuur te creëren en te manipuleren. Aan een *Presentation* kunnen *Slides* worden toegevoegd via append() methodes. Verder kunnen *Slides* worden opgevraagd via getSlideItem(int i). Voor *Slides* en *SlideItems* werkt het vergelijkbaar.

Presentation bevat daarnaast toestand en gedrag voor de vertoning van de presentatie op het scherm: het bevat een attribuut slidenumber, en methoden als nextSlide, prevSlide en setSlideNumber. Ook de methode getCurrentSlide hoort bij dit protocol.

3 De user-interface

De structuur van de user-interface van JabberPoint is relatief simpel (zie ook figuur 4). Het window op het topniveau wordt gerepresenteerd door de klasse *SlideViewerFrame*, dat weer een object *SlideViewerComponent* bevat. Deze component is verantwoordelijk voor het tekenen van de huidige *Slide* in de meegegeven *Presentation* in het venster. De initialisatie van deze structuur gebeurt in de main() methode van JabberPoint.



FIGUUR 4 De klassen voor de user-interface van JabberPoint

Met de *SlideViewerFrame* worden ook twee "controllers" geassocieerd: een *KeyController* en een *MenuController*. De *KeyController* luistert naar het toetsenbord en vertaalt bepaalde toetsaanslagen naar methode-aanroepen op het object van de klasse *Presentation* dat hem bij creatie is meegegeven (vanuit de initialisatie van JabberPoint en *SlideViewerFrame*). De *MenuController* creëert een menustructuur voor de *SlideViewerFrame*, en handelt menu-acties af door berichten te sturen naar de geassociëerde *Presentation*.

De berichten die vanuit de *Controllers* naar de *Presentation* worden gestuurd zijn typische "navigatie"-operaties: ga naar de volgende slide, ga naar de vorige slide, et cetera. Maar ook het laden van een presentatie (zie de open menu optie in *MenuController*) wordt gedaan door berichten te sturen naar *Presentation*.

De *SlideViewerComponent* is verantwoordelijk voor het tonen van de huidige slide uit de geassocieerde *Presentation*. Hiertoe luistert de *SlideViewerComponent* naar de *Presentation* voor veranderingen. In *Presentation* is ingebouwd dat veranderingen in de "current slide" of de toestand van de *Presentation* worden doorgegeven aan de (precies één) *SlideViewerComponent* die hij kent. Dit roept natuurlijk om generalisatie via een Observer-patroon (zie latere inwerkopgaven; de originele JabberPoint had hier ook een "observer" ingebouwd).

4 Het tekenen van Slides

Het daadwerkelijk tekenen van de inhoud van een *Slide* op het scherm vindt plaats in de methode paint van *SlideViewerComponent*. Let wel: de *Slides* worden getekend op de GraphicsContext van de *SlideViewerComponent*.

De methode paint vult het venster met een achtergrondkleur, zet een tekstje met het volgnummer van de *Slide* op het scherm, en delegeert vervolgens naar de methode draw van de actuele *Slide*. Daarbij worden als parameter de GraphicsContext, de originele component (als ImageObserver, voor het laden/tekenen van plaatjes) en de beschikbare oppervlakte voor het tekenen van de *Slide* meegegeven.

Om *Slides* goed te kunnen weergeven op verschillende oppervlaktes wordt bij het tekenen van de *SlideItems* op een *Slide* rekening gehouden met een schaalfactor. Deze wordt berekend door de beschikbare oppervlakte te vergelijken met een referentiegrootte voor *Slides*. De relevante constanten zijn referenceWidth en referenceHeight die worden gedeclareerd in de klasse *Slide*. De methode getScale berekent de schaalfactor die nodig is bij het tekenen.

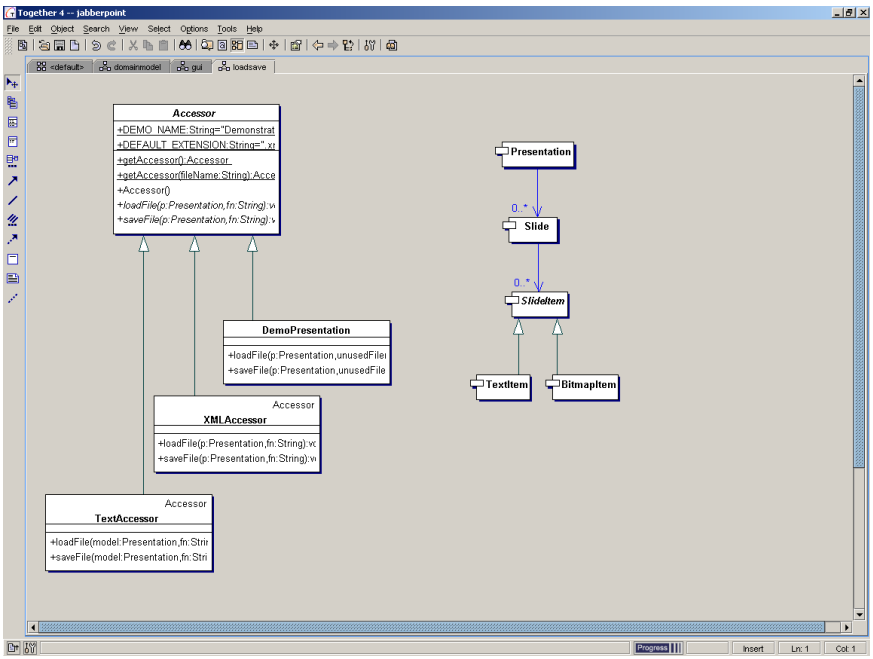
Het tekenen van een *Slide* bestaat vervolgens simpelweg uit het (via delegatie) tekenen van de titel (die tijdelijk tot *TextItem* wordt omgezet) en de diverse *SlideItems* op de *Slide*. Om de juiste verticale positie te bepalen wordt daarbij ook de boundingbox van de *SlideItems* opgevraagd (die eigenlijk alleen de breedte en de hoogte oplevert). Bij het aanroepen wordt ook de *Style* van het betreffende *SlideItem* meegegeven, op basis van het level.

Het tekenen van een *BitmapItem* is simpel: er wordt een Image gecreëerd op basis van de filenaam, en dit wordt, op de juiste schaal en op de juiste positie, getekend.

Het tekenen van een *TextItem* is wat subtieler. Lange teksten moeten worden afgebroken en verdeeld over meerdere regels. Hiertoe wordt de tekst als AttributedString gerepresenteerd (die ook mogelijkheden voor opmaak biedt) en worden afbreekpunten gevonden.

5 Het laden/bewaren van presentaties

Het laden en bewaren van presentaties gebeurt door de klassen in de *Accessor*-hiërarchie. Deze klasse definiëert twee methoden, een voor het laden en een voor het bewaren van een presentatie. Subklassen zijn bedoeld voor het laden/bewaren van presentaties in verschillende formaten. De klasse *Accessor* biedt verder een Factory-methode om de juiste subklasse te instantiëren op basis van de (extensie van een) meegegeven filenaam.



FIGUUR 5 Klassen voor het laden en bewaren van presentaties

Er zijn drie concrete *Accessors*, één voor XML (lezen en schrijven), één voor gestructureerde tekst (lezen) en een ingebouwde demonstratie presentatie-generator (lezen). Er zijn diverse demo-files te vinden met presentaties in verschillende formaten. Let wel: de XML-files moeten voldoen aan de DTD-structuur in jabberpoint.dtd.

De schrijvers lopen over de structuur van de domeinobjecten heen en mappen deze naar het juiste formaat. Op dezelfde manier wordt in de inlezers in één routine de hele structuur geinitialiseerd.

6 De inhoud van de bouwsteen

In deze bouwsteen vind je naast deze beschrijving, een zip met daarin de uitgangsversie van Jabberpoint voor deze opdracht.
Daarnaast bevat de bouwsteen de bibliotheken jdom.jar en xerces.jar.

OPDRACHT

Pas het systeem aan zodat slidetitels binnen JabberPoint als als TextItems gerepresenteerd worden. Je hoeft daarvoor niet de DTD aan te passen: bij het inlezen van een XML-bestand kunt u van een titel een textItem maken. Andere klassen mogen eventueel nog gebruik blijven maken van de methoden getTitle() en setTitle().