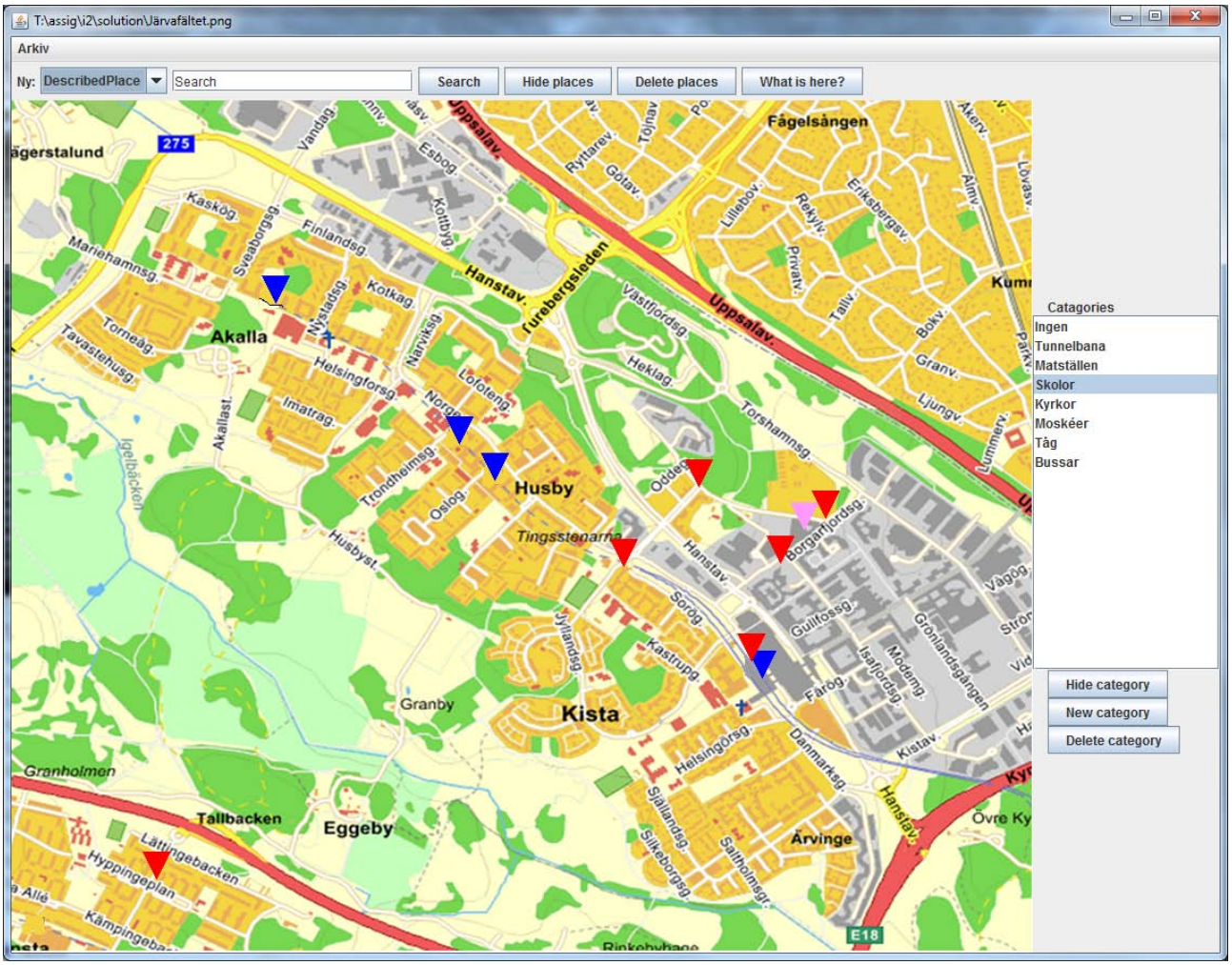
Inlämningsuppgift 2

Denna uppgift är den andra (av två) obligatoriska inlämningsuppgiften på delkursen PROG2 Programmering 2 VT2015. Uppgiften ska lösas enskilt eller i grupper om högst tre personer. Inlämningsinstruktion finns sist i det här dokumentet.

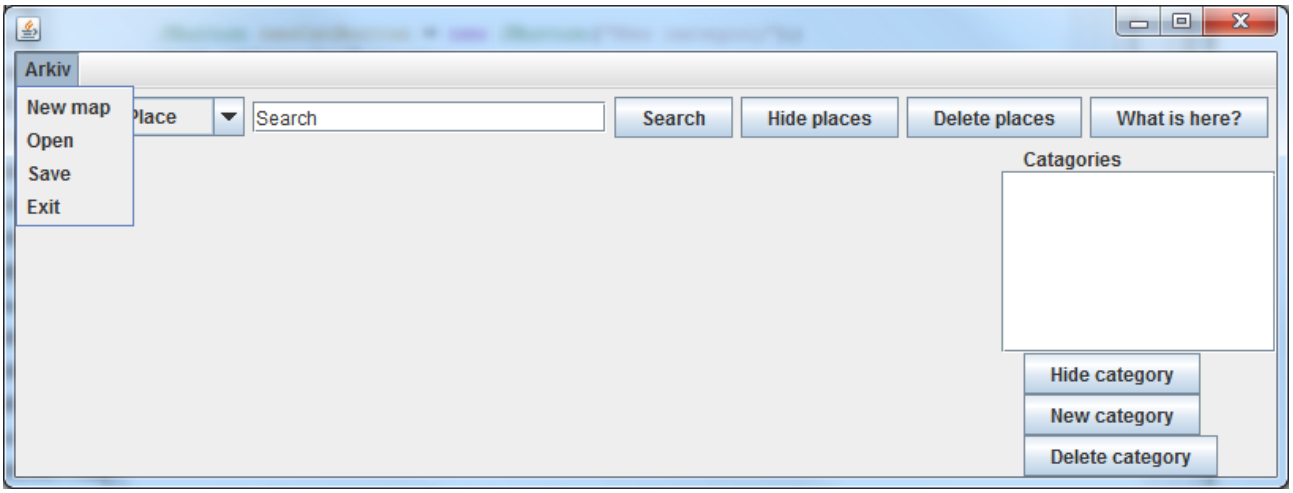
Uppgiften går ut på att skriva ett program som låter användaren öppna en karta (en bakgrundsbild) och med musklickningar definiera platser på kartan. Platsern har namn, kan ha en beskrivande text och hör till någon kategori (t.ex. Bussar, Tunnelbana, Matställe osv). Uppgiften är inspirerad av användargränssnittet på kartor.eniro.se, men är givetvis mycket förenklad och implementeras helt annorlunda: kartan är bara en bild, det finns ingen zoomning, man kan inte flytta på kartan (den är inte större än vad som visas osv).



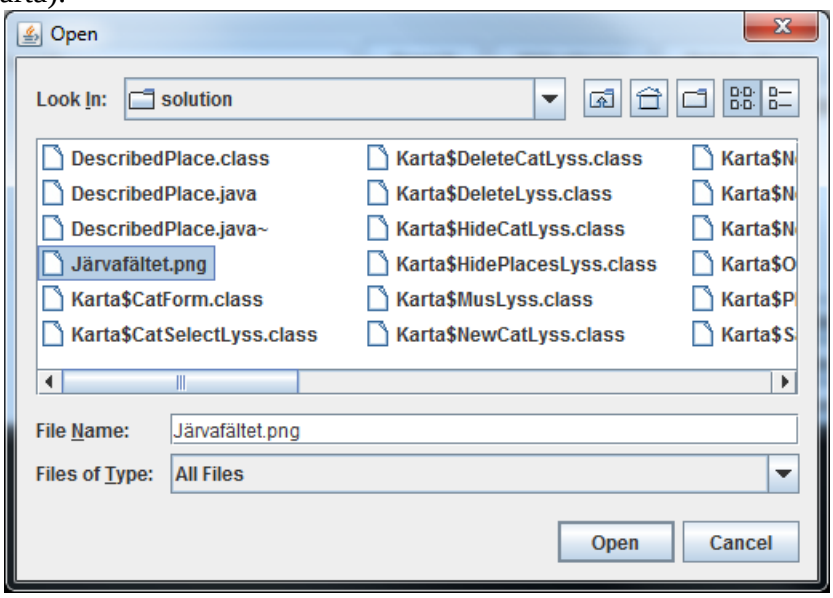
På ovanstående bild är det de färgade trianglarna som visar platser som användaren har definierat, tanken är att triangelns spets pekar på punkten (pixeln) där platsen är. Färgen på triangeln bestäms av vilken kategori som platsen tillhör, som i sin tur bestäms av vilken kategori som var vald när platsen skapades.

Kartan

Kartan är en vanlig bildfil.
När programmet startas visas ett fönster som kan se ut så här (visas med menyn Arkiv nerfält):



Om man väljer New map visas en fildialog där användaren kan välja en bild (förhoppningsvis föreställande en karta):



Bilden ska laddas in och fönstrets storlek anpassas så att hela bilden visas.

Position

Varje plats har ett namn som anges när platsen skapas, en position och en färg (som bestäms av kategorin). Positionen anger helt enkelt platsen pixelkoordinater och ska representeras med en egenskriven klass med namnet `Position`. Objekt av denna klass ska kunna användas som nycklar i hashtabeller för att snabbt kunna hitta vilken plats som ligger på en viss position.[1]

Platser

Platser är tänkta att kunna vara av olika typer, innehållande olika typer av information. I uppgiften ingår att det ska finnas två sorters platser: NamedPlace som bara har ett namn och DescribedPlace som förutom namnet har en beskrivande text. Namn och beskrivning matas in av användaren när hen skapar platsen och kan inte ändras sedan. Programmet ska vara förberett för att i en framtida utbyggnad ha fler platstyper, t.ex. platser som kan innehålla en bild, en tidtabell, öppettider osv.

[1] Det finns en liknande klass Point i Javas bibliotek, men för övnings skull är det alltså obligatoriskt att skriva en egen sådan klass istället.

En plats skapas genom att användaren väljer kategori i listan till höger och sedan platstypen i comboboxen i övre vänstra hörnet. Då ska markören över kartan ändras till ett kors (för att markera att nästa klick skapar en plats) och en klick på kartan skapar en plats på den klickade positionen. Obs att det är tänkt att den nedre triangelspet ska visa var platsen finns, så det behövs viss justering av koordinater för platsen. Efter att platsen är skapad ska inte kartan vara mottaglig för klickning förrän platstypen väljs i comboboxen på nytt. Om ingen kategori är markerad när en plats skapas får platsen ingen kategori och dess färg är svart.

När en NamedPlace skapas ska användaren ange platsens namn, vilken kan ordnas med en JoptionPane.showInputDialog. När en DescribedPlace skapas ska användaren dels ange namnet, dels en beskrivning som kan bestå av flera korta rader – för detta bör en JTextArea visas i en dialogruta, fast efter att dialogen är klar så sparas endast den inmatade strängen i Plats-objektet.

En Plats har tre olika tvåvärdiga tillstånd att hålla reda på: den kan vara hopfälld (visas som en triangel) eller utvikt (visas med sitt namn eller beskrivning), den kan vara markerad eller ej och den kan vara synlig eller gömd (man kan gömma platser man inte är intresserad av för att kartan inte ska bli oöverskådlig).

Hantering av om platsen är synlig eller ej kan implementeras med Swings grafiska komponenters `setVisible`-metoder (Plats-klasser bör givetvis ärva från `JComponent`). Om platsen visas hopfälld eller utvikt resp. markerad eller ej måste implementeras i Plats-klasserna.

Användaren ska kunna markera/avmarkera en plats genom att kicka på den med vänster musknapp[2]. Flera platser ska kunna vara markerade samtidigt. Hur en markerad plats visas kan du bestämma själv, i exemplet nedan är markerade platser omgivna med en röd rektangel (den vänstra röda triangeln, den nedre blå triangeln och den utvikta översta platsen.

Man ska kunna vika ut eller fälla ihop platser genom att klicka på dem med höger musknapp. Nedan till vänster är alla platser hopfällda, till höger har den översta röda platsen (tydligen en NamedPlace) vikts ut (texten 514,516, tänkt som bussnr på en busshållplats), och den rosa triangeln som markerar NOD-huset (tydligen en DescribedPlace) har vikts ut med sin beskrivning. En högerklick på de utvikta platserna fäller ihop dem igen.

[2] I en MouseListener-metod kan man kolla vilken knapp som trycktes på så här:
if (mev.getButton() == MouseEvent.BUTTON1)… där mev är MouseEvent-argumentet. Det finns fördefinierade namn för BUTTON1, BUTTON2 och BUTTON3.

Obs! att för att kunna rita den utvikta informationen måste man utvidga Plats-komponentens Bounds (med `setBounds`-metoden). När Platsen fälls ihop igen borde Bounds återställas så att klickningar med musen utanför triangeln inte markerar platsen.

Obs vidare! att det inte ställs några högre krav på presentation av informationen om platser i utvikt läge. Lägg inte ner tid på det (anpassningar till olika typsnitt och storlekar, korrekt radbrytning av för långa rader osv), bara informationen presenteras så kan detaljer fixas till om tiden tillåter.

Programmet ska kunna hålla reda på vilka platser som är markerade – operationerna Hide places och Delete places arbetar på de markerade platserna.

Search
Operationen Search hämtar strängen som matats in i sökfältet, letar upp och markerar alla platser som har denna sträng som namn och gör dem synliga och markerade. Obs att flera platser kan ha samma namn. Sökresultatet visas alltså genom att Platserna markeras. Om andra Platser var markerade innan sökningen så avmarkeras de.
Search-operationen förutsätter att man snabbt kan söka upp alla platser som har ett visst namn. Sekvensiell genomsökning av olämpliga datastruktur accepteras inte.

Hide places
Operationen Hide places gömmer alla markerade platser och gör dem avmarkerade.

Delete places
Oprationen tar bort alla markerade platser (inte bara så att de inte syns på kartan, utan objekten ska tas bort från alla datastrukturer där de kan finnas).

What is here
Användaren ska kunna klicka på kartan för att få veta om det finns någon osynlig Plats på denna position. Om det finns en plats där så ska den visas, om det inte finns så ska ingenting hända.

Obs att komponenter som inte syns inte reagerar på musklickningar. Det behövs, liksom i samband med Search, någon datastruktur där man snabbt ska kunna få fram om det finns en plats på den klickade positionen eller inte.

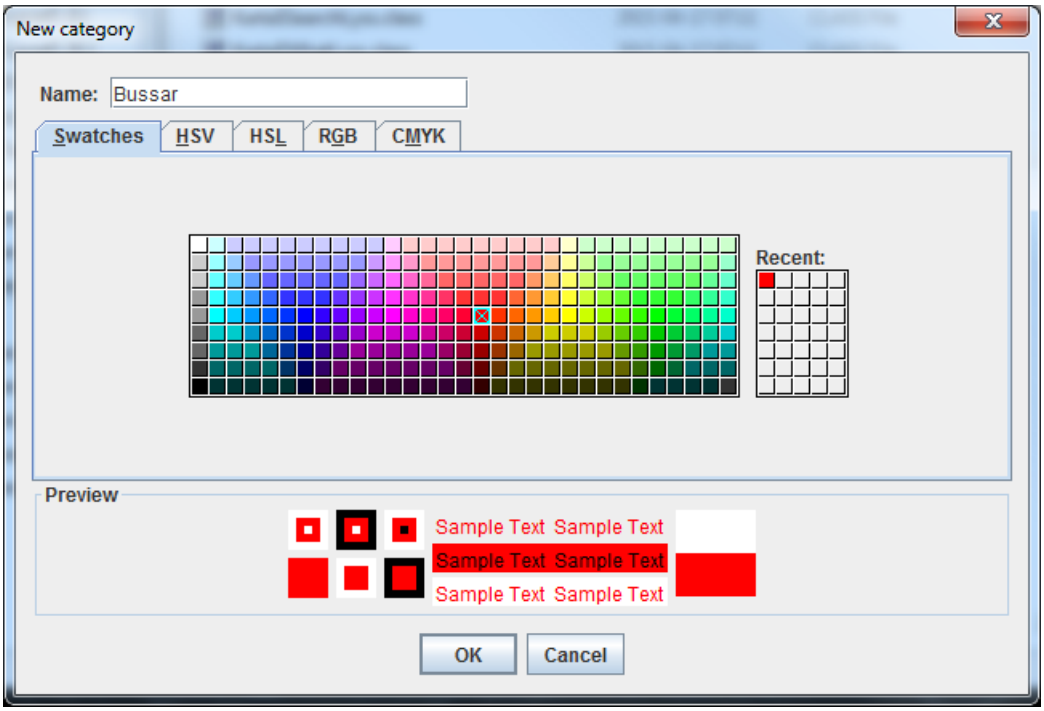
Obs vidare att det kan vara väldigt svårt att träffa just den pixeln där den osynliga platsen är. Man ska därför undersöka inte bara den position där användaren har klickat, utan alla positioner i en omgivning till klickpunkten, i en ruta på 7x7 positioner där klickpunkten är den mittersta positionen.

Kategorier
Platserna kan som sagt tillhöra en kategori (även om det kan finnas kategorilösa platser). Kategorierna visas i listan i högra panelen. Platsen tillhör den kategori som är vald när platsen skapas, om ingen kategori är vald då så blir platsen kategorilös. Till varje kategori hör en färg, platsens färg (som triangeln ritas ut i i hopfällt läge) är kategorins färg. Kategorilösa platser är svarta. Meningen med kategorierna är att man ska kunna visa, gömma eller ta bort alla platser som hör till en viss kategori, t.ex. alla T-baneingångar, eller busshållplatser, matställen osv.

Hantering av platser i en kategorier
Platser som hör till en kategori ska visas när användaren markerar den kategorin i listan. Däremot ska platser hörande till en viss kategori inte gömmas när kategorin blir avmarkerad. Vill man gömma alla platser som hör till en kategori så klickar man på knappen Hide category – platser som hör till den valda kategorin görs osynliga. Man kan även ta bort en hel kategori – alla platser som hör till kategoring och kategorin själv tas bort.

Skapande av kategorier

Det finns inga kategorier från början, anändaren skapar kategorier genom att klicka på knappen New category. Sedan ska man kunna mata in kategorins namn och färg. Hur den dialogen ska se ut, speciellt färginmatningen, kan du bestämma själv (men se förslaget nedan). Resultatet av inmatningen ska vara ett objekt av standardklassen `Color`.
Det är alltså ok att ha JRadioButtons med olika färger, textinmatning eller annat, men det finns ett förberett sett för detta i Swing, nämligen användning av en `JColorChooser`. Nedan exempel på en sådan instoppad i en panel med ett textfält överst och presenterad i en dialogruta med metoden `JoptionPane.showConfirmDialog`.



`JColorChooser` är inte beskriven i Skansholm, men finns beskriven i Lewis&Loftus, och framför allt i The Java Tutorial "How to Use Color Choosers" på https://docs.oracle.com/javase/tutorial/uiswing/components/colorchooser.html

Arkiv-menyn

Programmets data ska kunna sparas på en fil[3] och laddas in vid nästa körning. Operationer för detta finns i en menyn Arkiv i menybalken (du får gärna ändra namnet på menyn till Archive, eller tvärtom byta ut de engelskspråkiga texter på knapparna osv till svenska). Operationerna i menyn är New map, Open, Save och Exit.

Open ska visa en fildialog för att välja ut och öppna en fil som skapats av programmet vid en tidigare körning. Det kan vara lämpligt att bestämma sig för någon viss ändelse för filnamnet, t.ex. ".krt" för "karta" och filtrera de filer som visas i dialogrutan (detta är dock inte obligatoriskt). Om ni bestämmer er för någon sådan ändelse borde ni helst se till att den inte kolliderar med ändelser för andra program ni använder.

Save ska bete sig på två olika sätt beroende på om det aktuella "dokumentet" är namngivet eller inte[4]. "Dokumentet" är inte namngivet om det har skapats genom New map och inte sparats, har det

[3] Filhantering gås igenom på föreläsning 16
[4] Om du vill kan du göra två operationer. Save och Save as

däremot sparats tidigare eller öppnats från en fil så är det namngivet. Om "dokumentet" inte är namngivet så ska Save öppna en fildialog för val av filnamn att spara på och därefter spara data, annars sparas data utan någon mer dialog med användaren.

Exit ska avsluta programmet. Programmet ska även kunna stängas via programmets stängningsruta i övre högra hörnet. Om det finns osparade förändringar ska det visas en dialogruta som varnar om att det finns osparade förändringar och frågar om man ändå vill avsluta – användaren har då möjligheten att avbryta operationen.

Obs att både New map och Open kan väljas även när man har en karta inladdad och platser och kategorier skapade. I så fall måste det aktuella "dokumentet" avslutas innan det nya kan användas. Användaren bör ges samma varning som vid Exit i fall det finns osparade förändringar. Dessutom måste alla datastrukturer som innehåller platser, kategorier osv tömmas, så att de nya kan skapas/laddas in.

Inlämning

Inlämningen ska göras senast söndagen 2015-05-03 för rättning i samband med kursen. Inlämning av uppgiften sker på kurssidan i ilearn2.dsv.su.se.
Det som ska lämnas in är en exekverbar JAR-fil[5] innehållande lösningen. JAR-filen ska innehålla en mapp med namnet `sources` innehållande källkoden till lösningen.

För att ladda upp JAR-filen klicka på “Inlupp 2 - inlämning” under rubriken “Inlämningsuppgift 2”. Klicka på kommentarsfältet för Submission comments och mata in följande information:
- namn och personnummer för samtliga gruppmedlemmar
- på vilket operativsystem (Windows, Linux, MacOS) programmet har utvecklats på
Klicka sedan på knappen “Add submission”. Dra in JAR-filen. Klicka på knappen "Save changes". Om uppgiften löstes i grupp är det bara en i gruppen som ska lämna in.

Om du inte hinner bli klar till den angivna deadlines så kommer ett andra tillfälle den 14 augusti 2015, med handledning under vecka 33.

Om du inte hinner bli klar med uppgiften till deadlinen i augusti så upphör uppgiften att gälla och man ska lösa inlämningsuppgifterna på en kommande omgång av kursen. Obs att detta gäller samtliga delar av inlämningsuppgiften, alltså även inlupp 1[6]. Nästa omgång av PROG2 ges på vårterminen 2016.

[5] JAR-filer gås igenom på föreläsning 17.
[6] inlupp 1 utgör inte ett eget examinationsmoment utan är en del av examinationsmomentet Inlämningsuppgift 4,5hp..