DEPARTMENT OF INFORMATION ENGINEERING
UNIVERSITY OF PADOVA

Università degli Studi di Padova

Facoltà di Ingegneria

Corso di Laurea in Ingegneria Informatica

Tesi di laurea

Gestione mappe storiche tramite software touch screen

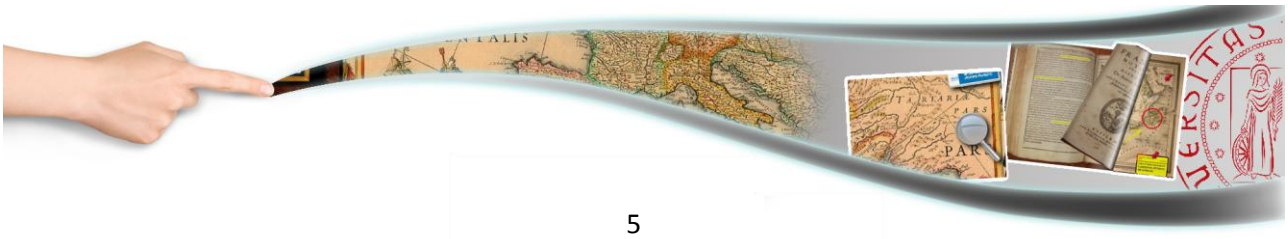
Relatore: Prof. Federico Filira

Laureando: Olga Zakharova

Matricola N. 564071-IF

Anno accademico 2010 - 2011

Indice



1. Introduzione

L'obiettivo di questa tesi è stato quello di realizzare un'applicazione multitouch che offrisse all'utente la possibilità di esplorare mappe digitali in maniera facile ed intuitiva utilizzando esclusivamente le dita. Questo strumento è pensato per mappe, libri antichi, cataloghi, documenti etc., tutto navigabile con le dita. Inoltre permette di scegliere l'immagine desiderata, spostarla, zoomarla, disegnare/scrivere su di essa, cancellare il disegno e sfogliarne i diversi contenuti. Ideale per ambienti aperti al pubblico quali musei, biblioteche, fiere, aeroporti, centri commerciali, alberghi, showroom.

Questo elaborato è stato realizzato per i musei (come contenuti ho scelto le mappe storiche) con l'obiettivo di permettere la visualizzazione e navigazione di mappe storiche digitalizzate con collegati documenti e video relativi ai diversi eventi storici.

2. Tecnologia

Hardware:	HP TouchSmart IQ512it - 22"
SO:	Microsoft Windows 7
Linguaggio:	Microsoft C# + WPF
Framework:	Microsoft .Net 3.5 sp1
Ambiente di sviluppo:	Microsoft Visual Studio 2010 Professional
Multitouch:	Libreria NextWindow

3. Interfacce grafiche interattive

3.1 Un po' di storia

In un primo periodo l'evoluzione dei calcolatori riguardò il lato architetturale e il loro funzionamento interno, fase che si concluse con la massiccia adozione da parte dei produttori dell'architettura di von Neumann. Le successive evoluzioni dei computer riguardarono sia il 'motore' della macchina sia la sua interfaccia grafica ovvero il luogo e le modalità con cui l'utente fornisce input al sistema e riceve feedback in output dallo stesso.

Dalle prime interfacce costituite esclusivamente da interruttori si è arrivati ad interfacce utente grafiche (GUI) gestite tramite mouse e tastiera, passando per schede perforate e terminali.

Le future interfacce per i nostri sistemi di computazione diverranno sempre più naturali, cioè sfrutteranno la potenzialità delle modalità di interazione da noi già normalmente usate nel quotidiano e nell'interazione uomo-uomo, come ad esempio l'utilizzo della voce e delle gestualità del corpo. Le interfacce future saranno quindi di facile utilizzo anche per chi è estraneo al mondo del computing e saranno sempre più presenti nell'ambiente che ci circonda.

Il primo dispositivo moderno con tecnologia **touchscreen** è stato immesso sul mercato agli inizi degli anni '70 quando si videro le prime applicazioni con sensori tattili nel monitor di un computer. L'HP-150 del 1983 è stato il primo computer con tecnologia touchscreen. Il primo dispositivo touchscreen con cui molte persone hanno familiarizzato è stato il cellulare e nello specifico il modello **iPhone** (Apple Inc.) uscito nel 2007. Oggi il touchscreen lo ritroviamo un po' ovunque: computers, cellulari, tablets, videogiochi, navigatori satellitari, orologi, macchinari industriali, dispositivi militari, etc.

3.2 Le librerie principali

OpenGL (**Open G**raphics **L**ibrary) è una specifica che definisce un'API multipiattaforma per scrivere applicazioni che utilizzano grafica 2D e 3D, viene utilizzata prevalentemente in ambienti Unix. E' stata realizzata da Silicon Graphics nel 1992 ed è molto apprezzata in ambienti CAD, realtà virtuale, effetti speciali per film e programmi televisivi.

GTK+ (acronimo di **GIMP ToolKit**) è un insieme di strumenti di cui il principale è la libreria libgtk per la creazione di interfacce grafiche. Sviluppato in C, supporta nativamente l'ambiente grafico X Window System ed è utilizzato da molti ambienti desktop X Window per piattaforme Unix/Linux.

Qt è una libreria multipiattaforma che utilizza dei widget per la realizzazione di interfacce grafiche. Utilizza il linguaggio C++ ed è stato adottato da diversi ambienti desktop come KDE.

QuickDraw è la libreria di interfaccia grafica di Apple. Fa parte integrante del sistema operativo installato sui computer Machintosh. Nelle attuali versioni del sistema Mac OS X viene invece utilizzata la libreria Quartz che offre risultati più performanti.

GDI è la libreria standard del mondo Windows che offre medie prestazioni ma buona interoperabilità. E' stata nel tempo sostituita da GDI+ che garantisce migliori risultati e una migliore gestione dei dispositivi hardware moderni.

DirectX è la libreria del mondo Windows che offre elevate prestazioni grafiche, video e di animazione 3D. L'applicazione da me realizzata in WPF si basa su questa libreria per ottenere la miglior fluidità della grafica.

3.3 Lo stato dell'arte delle interfacce grafiche

3.3.1 Soluzioni commerciali

Dopo i cellulari touchscreen in questi ultimi mesi si stanno diffondendo i **tablet** che offrono un'interfaccia grafica con multitouch e dimensione del display di circa 10" come ad esempio **iPad** di Apple e **Galaxy** di Samsung che installa il nuovo sistema operativo **Android** di Google.

Relativamente a sistemi di dimensioni maggiori negli ultimi anni è stato fatto ampio uso, nei network televisivi italiani e non, di **Surface** della Microsoft prevalentemente per la presentazione della rassegna stampa e del meteo. Questo sistema si basa sul sistema operativo Windows 7 ed è programmabile in C# con ambiente grafico WPF.

3.3.2 Soluzioni del futuro

Realtà virtuale

La realtà virtuale è la tecnologia che permette di ottenere esperienze immersive grazie all'ausilio di dispositivi quali caschi, occhiali, tute, guanti, sensori, etc. Non ancora molto diffusa a livello commerciale permette un'interazione uomo-macchina realistica grazie ad interfacce grafiche tridimensionali.

AR Realtà aumentata (augmented reality)

La realtà aumentata permette di visualizzare oggetti multimediali sovrapposti ad oggetti reali. Attualmente viene utilizzata solo in ambiti molto specifici come il settore militare, medico e scientifico. Sovrappone graficamente agli oggetti reali informazioni virtuali come testo, immagini, video, mappe, modelli tridimensionali.

Wearable gestural interface

Dal punto di vista della ricerca ai massimi livelli segnalo **SixthSense**. E' un dispositivo indossabile con interfaccia gestuale che aumenta il mondo fisico con informazioni

digitali e consente agli utenti di utilizzare i movimenti naturali delle mani per interagire con tali informazioni. È stato sviluppato da un dottorando del Fluid Interfaces Group al MIT Media Lab.

http://www.youtube.com/watch?v=daXL_12miqA

4. Specifiche di progetto e metodologia di sviluppo

4.1 Specifiche del progetto

Il progetto è stato pensato per i musei, le specifiche erano le seguenti:

- organizzare le mappe per continente
- per ogni continente avere la lista delle mappe del continente prescelto
- muovere la mappa selezionata
- zoomare con uno o due dita la mappa selezionata
- disegnare/cancellare sulla mappa selezionata
- ingrandire nel punto desiderato (effetto lente d'ingrandimento)
- per ogni mappa allegare **n** punti interattivi e per ognuno di essi aggiungere una descrizione, **n** file video e **n** file pdf
- sfogliare le mappe come le pagine di un libro oltre che scorrere tutte le mappe a media dimensione.

4.2 Metodologia di sviluppo

Come metodologia di sviluppo ho seguito un procedimento **Agile**. La metodologia Agile è stata pensata per produrre velocemente codice funzionante focalizzandosi su ciò che il cliente considera di valore.

Le metodologie agili sono nate specificamente per l'ingegneria del software. Lo scopo è quello di rendere il software estremamente facile da modificare. In questo modo costa meno aggiornarlo con le necessità degli utenti, che cambiano frequentemente.

Gli approcci agili considerano il cambiamento come una necessità. Nelle metodologie di sviluppo tradizionali (Waterfall) l'analisi e il design sono fatti all'inizio del progetto e si cerca di prevedere i cambiamenti che potrebbero essere richiesti nel corso dello sviluppo come accade sovente.

Le metodologie agili invece propongono delle modalità di lavoro orientate al raggiungimento di piccoli risultati a breve termine, mantenendo il processo di sviluppo flessibile e pronto a considerare le inevitabili variazioni delle condizioni, dei requisiti e delle richieste dei clienti.

La metodologia si basa su rilasci brevi e iterazioni di sviluppo e test molto frequenti che permettono di costruire, a piccoli passi, un sistema complesso in modo incrementale.

5. Descrizione tecnica dell'applicazione

In fase di avvio l'applicazione controlla le sottocartelle della propria cartella **<Percorso installazione applicazione>\Mappe**. Ogni sottocartella viene considerata come un continente e ogni file di tipo immagine trovato al suo interno viene catalogato come mappa collegata al continente stesso. Inoltre per ogni mappa caricata viene verificata l'esistenza di un file di testo con lo stesso nome ma estensione txt da dove reperire le coordinate in pixel dell'evento storico, la relativa descrizione ed eventuali altri file tipo documenti pdf o video.

Ultimata la fase di avvio il software dispone quindi di una struttura di classi collegate tra loro che contengono le immagini miniaturizzate da usare nella toolbar e nel

menu, i riferimenti ai diversi file originali delle mappe e tutte le informazioni relative a coordinate e descrizioni.

L'applicazione permette quindi di selezionare un continente dalla toolbar semplicemente trascinandolo con le dita nella parte centrale. Oltre a presentare l'immagine originale del continente appare un menu a sinistra con tutte le immagini collegate. Sia la toolbar che il menu sono a scorrimento, cioè si spostano con le dita.

Sull'immagine selezionata, grazie ad una serie di strumenti, è possibile effettuare varie operazioni tra cui lo spostamento, lo zoom, scrivere/disegnare con colori diversi, cancellare o ingrandire un particolare tramite la lente. Per quanto riguarda lo zoom è stato implementato anche il multitouch che permette, con due dita, di ridimensionare il contenuto in maniera semplice ed intuitiva.

Oltre a navigare le mappe miniaturizzate del continente prescelto tramite il menu scorrevole verticale è stato implementato sia uno scorrimento orizzontale con visualizzazione delle immagini a media dimensione sia uno sfogliamento naturale delle pagine di un libro.

Per ottenere prestazioni grafiche ottimali le immagini miniaturizzate vengono caricate nella memoria della scheda video fin dalla fase iniziale mentre i contenuti originali vengono elaborati dinamicamente all'occorrenza.

Le operazioni di spostamento, ridimensionamento, ingrandimento con la lente e le aninimazioni sono state ottenute operando sulle trasformate degli oggetti stessi. La funzione di movimento è stata estesa per ottenere l'effetto inerzia successivo al rilascio dell'oggetto in movimento. La funzione d'ingrandimento multitouch tramite due dita è stata ottenuta calcolando il rapporto tra le diagonali della posizione corrente e precedente delle dita.

6. Interfaccia utente

Immagine N° 1 – Videata di esempio delle funzionalità

Questa videata rappresenta una mappa selezionata, il menu a scorrimento verticale a sinistra con le altre mappe collegate allo stesso continente, gli strumenti di lavoro nella parte destra, i punti interattivi completi di descrizione tramite i quali è possibile visualizzare i documenti e i video allegati nonché dei particolari evidenziati con lo strumento pennello.

Ad ogni mappa si possono associare **n** punti interattivi ognuno dei quali permette di visualizzare: una descrizione, **n** documenti pdf e **n** video.

7. Diagrammi

7.1 Diagramma classi applicazione

Le classi applicative gestiscono il funzionamento vero e proprio del programma. Per ogni unità logica è stata creata una classe specifica che contiene i dati relativi ai contenuti e le implementazioni funzionali.

7.2 Diagramma classi di supporto

All'interno delle classi di supporto vi sono varie funzioni, generalmente statiche, utilizzate per operazioni necessarie in varie aree del programma.

7.3 Diagramma classi di configurazione

Le classi di configurazione vengono utilizzate in fase di avvio del programma per inizializzare i parametri fondamentali dell'applicazione.

8. Fase di avvio del programma

La fase di avvio del programma è molto importante poiché è in questa fase che vengono caricati tutti i dati necessari all'esplorazione delle mappe.

Nella fase di avvio l'applicazione esplora tutte le sottodirectory contenute nella cartella **Mappe**. Le stesse vengono considerate come continenti e per ciascuna viene creata un'immagine principale ma di dimensioni inferiori rispetto alla mappa originale. Infine vengono caricate in nuove istanze della classe **MappaToolbar** che è un oggetto che estende la classe Image e contiene tutti i dati dello specifico continente. Ogni istanza viene aggiunta anche all'oggetto visivo **toolbar** per rappresentare l'icona del continente e inoltre il suo riferimento viene aggiunto ad una lista statica di classi di tipo **MappaToolbar**.

In ogni directory del singolo continente vengono caricati anche tutti i percorsi delle immagini presenti creando delle istanze della classe **MappaCategoria**. Inoltre se esistono dei file di testo che hanno lo stesso nome dell'immagine ma estensione txt, ne viene letto il contenuto e aggiunto ad una lista di classi **PuntiMappa** referenziata da ogni singola istanza di **MappaCategoria**. Questi file contengono i dati relativi ai Punti Interattivi, la descrizione dell'evento e gli eventuali nomi dei file di tipo documento o video.

Infine ogni istanza di **MappaCategoria** (mappa) viene aggiunta alla relativa lista contenuta in ogni istanza di **MappaToolbar** (continente).

I dati del documento txt sono organizzati nel seguente modo:

- Coordinata X del punto
- Coordinata Y del punto
- Testo che descrive l'evento storico
- Percorsi dei video allegati al punto
- Percorsi dei documenti allegati al punto

Per diversificare i percorsi dei video dai percorsi dei documenti viene preposta al nome del file una stringa di quattro caratteri che identifica il tipo di contenuto,

rispettivamente “**vid:**” e “**doc:**”. All’interno di ogni riga i campi sono separati dal simbolo “**;**”.

8.1 Esempio di file di dati

```
356,6;356,7;Distruzione Aquileia;doc:Dinastie_Valentiniana.pdf;vid:Valentiniano_I.flv
478,6;514,87;Battaglia del Reno;vid:vandali.flv;doc:alari.pdf
534,67;157,45;Treviri e’ incendiata dai barbari;vid:costa_barbari.flv;vid:treviri.flv
578,34;267,6;Re dei Visigoti;vid:alarico.flv;doc:re_visigoti.pdf
643,78;357,45;Sacco di Roma;vid:sant_agostino.flv;doc:arcadio.pdf;doc:teodosio_II.pdf
```

8.2 Procedura di caricamento dati

```
// Cerco tutti i file della directory, creo le immagini MappeToolbar e le aggiungo alla
// lista di tipo MappaToolbar, inoltre aggiungo le MappeToolbar al canvas cnvToolbar
// che rappresenta l’oggetto visivo Toolbar nella parte inferiore dello schermo
DirectoryInfo dirCorrente = new DirectoryInfo(Globali.percorsoMappe);
foreach (DirectoryInfo directory in dirCorrente.GetDirectories())
{
    // Creo la mappa principale del continente
    MappaToolbar mappaToolbar = new MappaToolbar(directory, cnvToolbarMappe);

    // Aggiungo alla lista lstMappeToolbar la mappa principale del continente
    Globali.lstMappeToolbar.Add(mappaToolbar);

    // Aggiungo la mappa MappaToolbar (di dimensioni inferiori) alla toolbar
    cnvToolbarMappe.Children.Add(mappaToolbar);

    // All’interno della classe del continente appena creata carico tutti i nomi dei
    // file reperiti nella relativa sottodirectory
    foreach (FileInfo currFile in directory.GetFiles())
    {
        // Processo il file solo se è di tipo immagine (jpg, png, bmp)
        if (Globali.IsImage(currFile.Name))
        {
            int posPuntoExt = currFile.Name.IndexOf(".");

            // Valorizzo il nome dell’eventuale file di testo
            nomeFilePunti = currFile.DirectoryName + "\\" + currFile.Name.Substring(0,
                posPuntoExt) + ".txt";

            // Aggiungo alla lista delle mappe del continente la nuova mappa
            mappaToolbar.lstMappeCategoria.Add(new MappaCategoria(currFile.FullName,
                nomeFilePunti, directory.Name));
        }
    }
}
```

8.3 Procedura di inserimento contenuti

Per caricare nuovi contenuti dopo aver installato l'applicazione basta copiare nella cartella del relativo continente le immagini delle mappe. Se ci sono punti interattivi bisogna creare un file di testo con lo stesso nome dell'immagine ma estensione txt con le coordinate dei punti, la descrizione e gli eventuali nomi dei file video e/o documenti (seguire l'esempio descritto sopra). Ogni video o documento elencato nel file di testo deve essere presente nella stessa directory.

9. Struttura dati

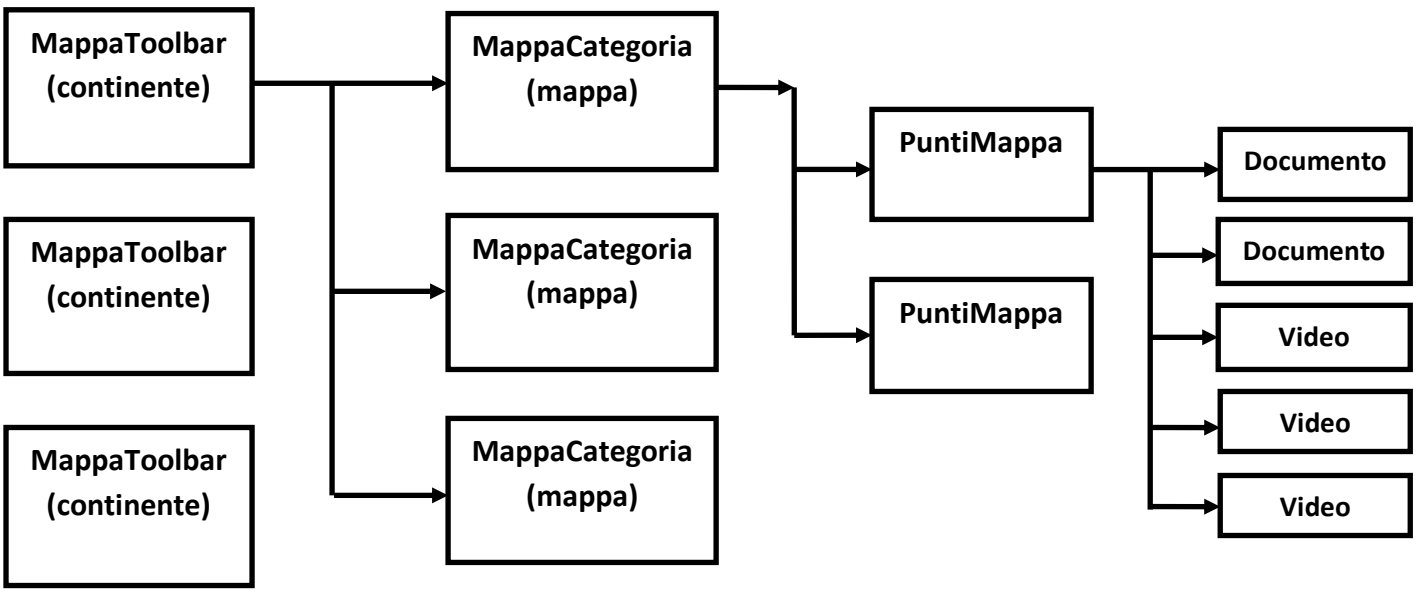


Immagine N° 2 – Collegamento logico delle classi fondamentali

Una volta caricata la struttura di classi risulta molto semplice reperire le informazioni neccessarie quando l'utente seleziona un continente o una specifica mappa. Ad ogni selezione viene aggiornata la mappa centrale con il continente prescelto, viene aggiornato il menu scorrevole di sinistra con tutte le mappe in dimensione ridotta, viene aggiornato lo scorrimento orizzontale con le immagini in dimensione media e infine vengono aggiornate le pagine del libro da sfogliare.

10. Classi principali

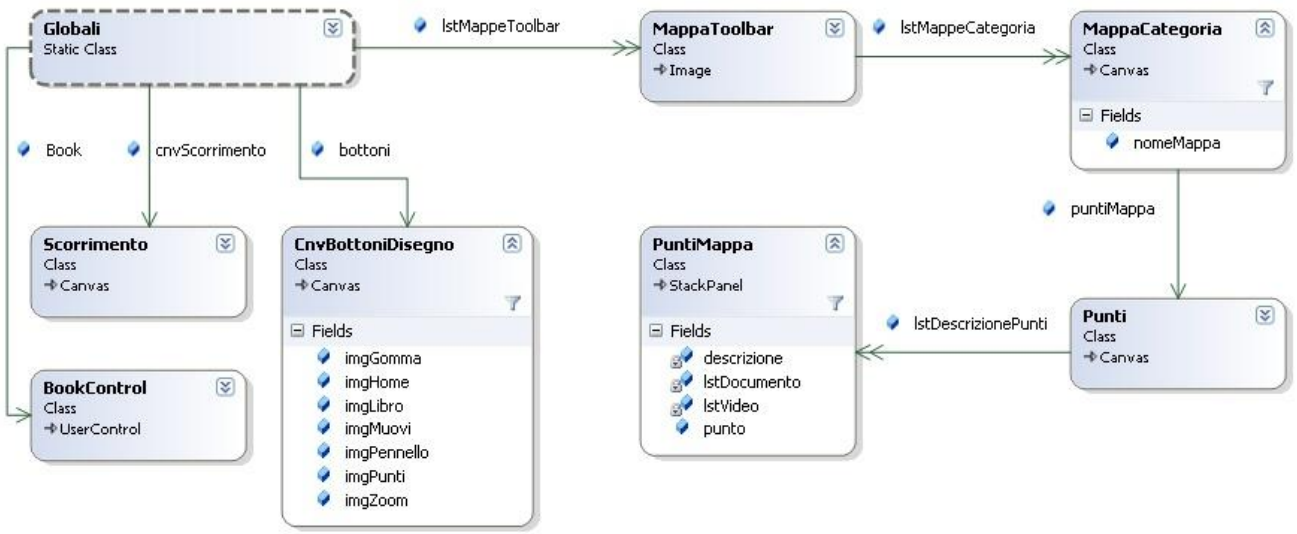
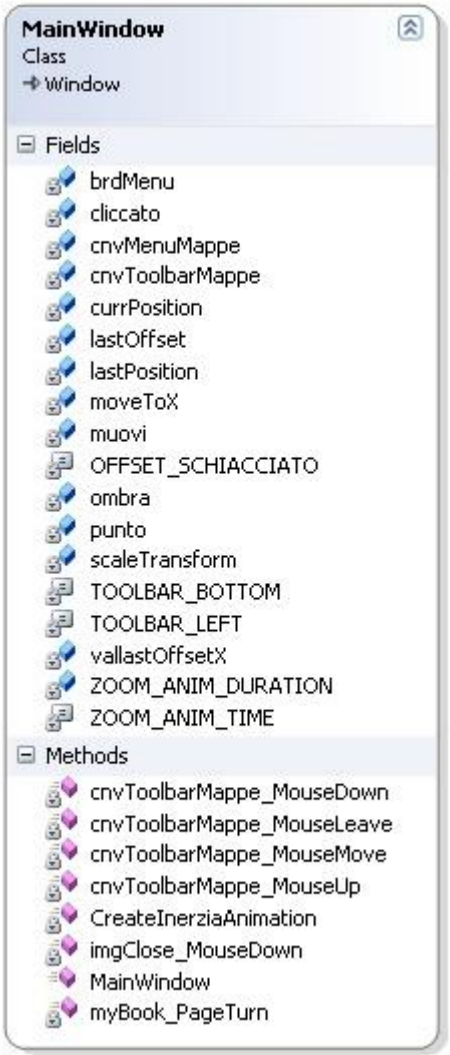


Immagine N° 3 – Collegamento delle classi fondamentali

```
// Riferimento al canvas contenitore dell'applicazione
public static Multitouch canvasMain;

// Riferimento alla mappa centrale selezionata
public static Canvas mappaCentrale;

// Riferimento alla mappa centrale con gli strumenti
public static Canvas cnvMappaeStrumenti;

// Riferimento al menu scorrevole verticale di sinistra
public static Canvas menu;

// Riferimento alla toolbar inferiore
public static Canvas toolbar;
```

Immagine N° 4 – Selezione mappa

10.1 Classe MainWindow



La classe **MainWindow** rappresenta la finestra principale che contiene tutte le altre.

Nel costruttore di **MainWindow** vengono caricate la toolbar con le immagini dei continenti, il menu scorrevole verticale, l'area principale di lavoro con tutti gli oggetti necessari e i bottoni degli strumenti. Inoltre vengono letti i file dalle directory e viene popolata la lista per il caricamento delle mappe nel menu scorrevole verticale oltre che le liste dei Punti Interattivi.

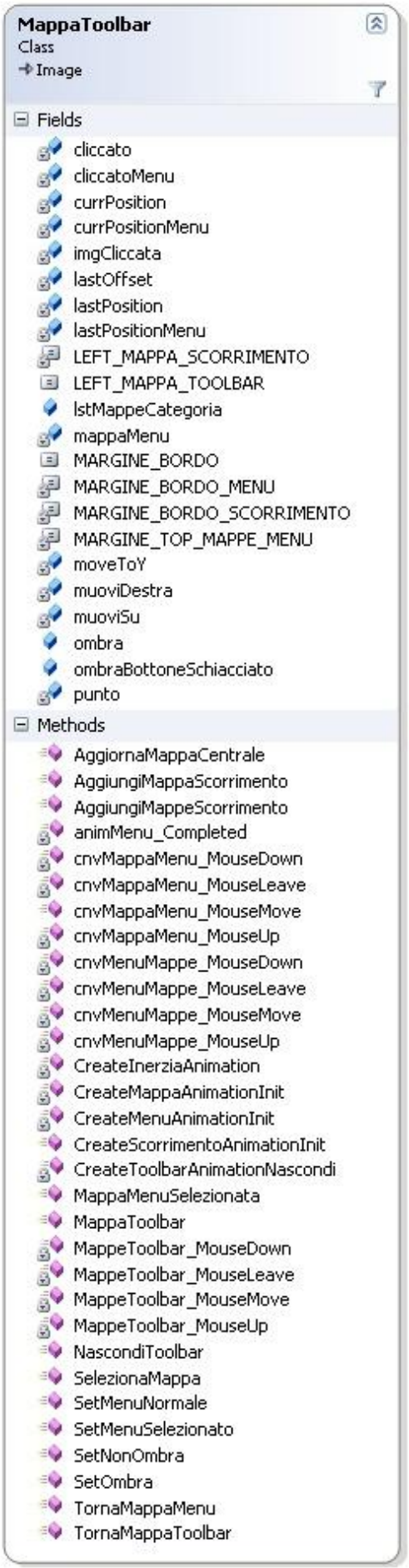
Nella funzione **cnvToolbarMappe_MouseMove** collegata all'evento MouseMove viene gestito lo spostamento orizzontale della toolbar con le immagini dei vari continenti. Inoltre sono stati gestiti anche i limiti dei bordi destro e sinistro.

La funzione **cnvToolbarMappe_MouseUp**, collegata all'evento MouseUp, gestisce il distacco del dito dallo schermo e muove autonomamente la toolbar imprimendole una forza d'inerzia.

Funzione di creazione dell'animazione inerzia:

```
/// <summary>
/// Funzione che torna l'animazione Inerzia
/// </summary>
/// <param name="toValue">Valore del punto di arrivo</param>
/// <returns>Animazione</returns>
private DoubleAnimation CreateInerziaAnimation(double toValue)
{
    var animCreateInerzia = new DoubleAnimation(toValue, ZOOM_ANIM_DURATION,
        FillBehavior.HoldEnd);
    DoubleAnimation.SetDesiredFrameRate(animCreateInerzia, Globali.FRAMERATE_DESIRED);
    animCreateInerzia.AccelerationRatio = .01;
    animCreateInerzia.DecelerationRatio = .5;
    animCreateInerzia.Freeze();
    return animCreateInerzia;
}
```

10.2 Classe MappaToolbar



La classe **MappaToolbar** è responsabile delle principali operazioni: selezione mappa dalla toolbar, aggiornamento dell'area di lavoro con la mappa selezionata, aggiornamento del menu a scorrimento verticale, aggiornamento del menu a scorrimento orizzontale con immagini a dimensione media, aggiornamento pagine del libro da sfogliare. Inoltre in questa classe vengono gestiti gli eventi di spostamento del menu a scorrimento verticale di sinistra.

La gestione dell'evento **MappaToolbar_MouseMove** gestisce il trascinamento verticale dell'immagine del continente prescelto.

La funzione **MappaToolbar_MouseUp** gestisce l'evento di rilascio dell'immagine del continente che, oltre a richiamare la funzione di aggiornamento continente **AggiornaMappaCentrale** di seguito descritta, utilizza il metodo **NascondiToolbar** per far scomparire la toolbar fuori dallo schermo.

La funzione **AggiornaMappaCentrale** aggiorna l'area di lavoro centrale con l'immagine del continente caricandola nelle sue dimensioni originali, carica il menu scorrevole verticale con le mappe del continente prescelto e aggiorna le pagine del libro e del menu a scorrimento orizzontale riportando ambedue alla posizione iniziale.

Il metodo **cnvMenuMappe_MouseMove** gestisce l'evento MouseMove del menu scorrevole verticale che permette lo spostamento verticale controllando che esso non esca dai bordi superiore e inferiore.

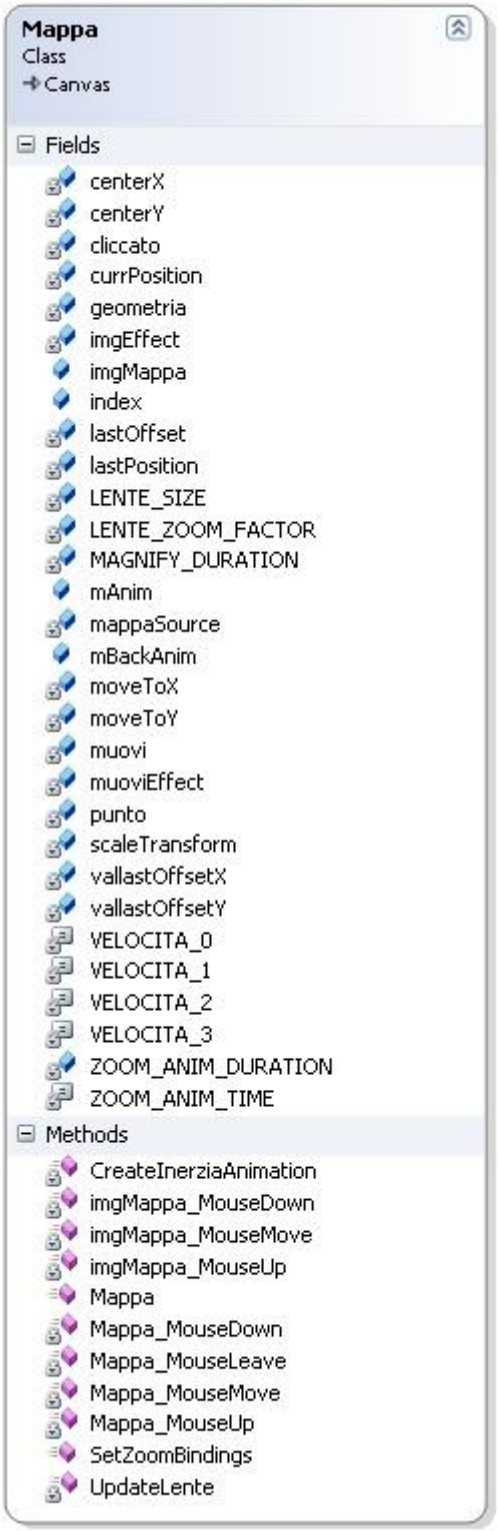
La funzione **cnvMenuMappe_MouseUp** gestisce l'inerzia del menu scorrevole. Anche in questo caso il movimento avviene controllando i bordi superiore e inferiore.

La funzione **AggiungiMappeScorrimento** aggiunge tutte le mappe del continente selezionato al canvas **Scorrimento** in modo che la mappa selezionata si trovi in mezzo all'area di lavoro. Questa funzione ottiene il risultato voluto richiamando ciclicamente il metodo **AggiungiMappaScorrimento** per ogni mappa del menu.

Analogamente a quanto avviene per la toolbar dei continenti è possibile selezionare una delle mappe dal menu scorrevole verticale. Infatti con uno spostamento orizzontale della mappa prescelta e successivo rilascio viene gestito l'aggiornamento dell'area di lavoro principale.

Immagine N° 5 – Selezione di una mappa specifica

10.3 Classe Mappa



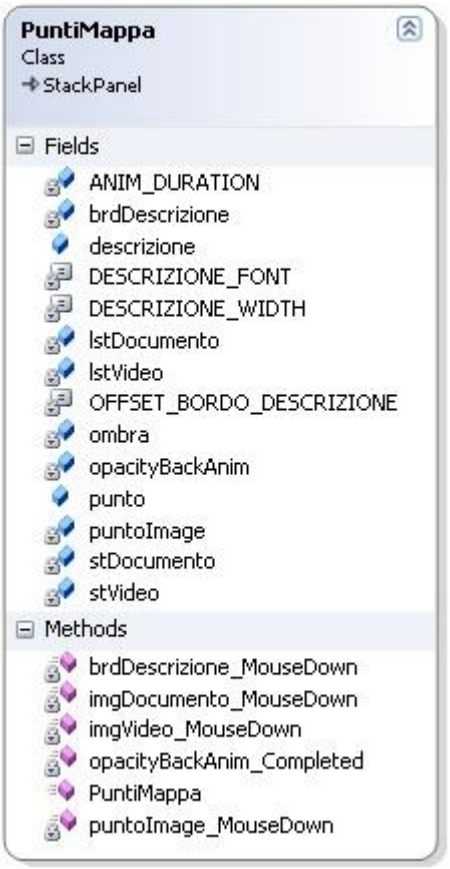
La classe **Mappa** è responsabile del caricamento dell'immagine nell'area di lavoro, del caricamento del canvas per poter **muovere** la mappa centrale, del caricamento dell'oggetto di tipo **inkCanvas** per poter disegnare/scrivere/cancellare sulla mappa oltre all'aggiornamento dell'oggetto **lente d'ingrandimento**.

La funzione **Mappa_MouseMove** gestisce l'evento MouseMove dell'immagine centrale che sposta la mappa, controllando i bordi destro, sinistro, superiore e inferiore in modo che l'immagine non esca dall'area di lavoro.

La funzione **Mappa_MouseUp** gestisce l'evento MouseUp della mappa centrale dell'area di lavoro, calcola la velocità del movimento e il nuovo punto di arrivo controllando che l'immagine non esca dai bordi e successivamente lancia l'animazione che imprime l'inerzia chiamando il metodo **CreateInerziaAnimation**.

La funzione **imgMappa_MouseDown** gestisce l'evento MouseDown dell'effetto lente, in sostanza chiama la funzione **UpdateLente** la quale aggiorna la posizione della lente. Questo effetto se è attivato dal relativo strumento, fa apparire una circonferenza di raggio prestabilito con l'immagine ingrandita nella posizione di contatto del dito con lo schermo. La funzione **imgMappa_MouseMove** gestisce l'evento MouseMove che, sempre richiamando il metodo **UpdateLente**, aggiorna la posizione della lente d'ingrandimento. Il metodo **imgMappa_MouseUp** gestisce l'evento MouseUp che fa sparire l'effetto lente.

10.4 Classe PuntiMappa

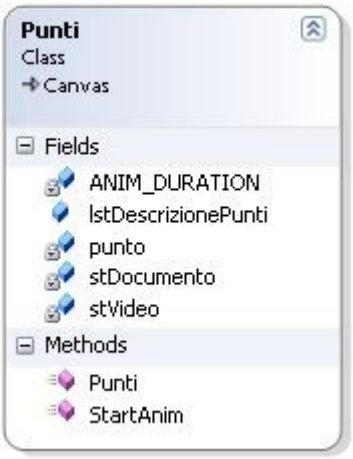


La classe **PuntiMappa** è responsabile del Punto Interattivo che viene aggiunto nella coordinata prestabilita. Inoltre al punto vengono aggiunti la descrizione dell'evento storico, i bottoni dei video e dei documenti allegati.

La funzione **imgVideo_MouseDown** gestisce l'evento MouseDown del bottone video il quale attiva il player selezionato e fa partire la visione del filmato.

Analogamente alla funzione precedente il metodo **imgDocumento_MouseDown** gestisce l'evento del bottone documento il quale attiva il visualizzatore di documenti pdf per permetterne la lettura.

10.5 Classe Punti



Nella classe **Punti** vengono letti i file di testo per ogni mappa e vengono aggiunti i **Punti Interattivi** con la relativa descrizione, i video e i documenti allegati.

10.6 Classe CnvBottoniDisegno

La classe **CnvBottoniDisegno** è responsabile della gestione degli strumenti.

La funzione **imgHome_MouseDown** gestisce l'evento MouseDown del bottone **Home** che visualizza la toolbar per dare la possibilità di scegliere un altro continente.

La funzione **imgMuovi_MouseDown** gestisce l'evento MouseDown del bottone **Mano** e permette di spostare la mappa centrale attivando il canvas **muoviCanvas** gestito nella classe **Mappa**. Inoltre è possibile ridimensionare l'immagine con due dita. Questa funzione viene gestita nella classe **Multitouch**.

Il metodo **imgPennello_MouseDown** gestisce l'evento MouseDown del bottone **Pennello** che attiva l'**inkCanvas** (layer di disegno), inoltre permette di cambiare il colore del Pennello.

Analogamente al metodo precedente, la funzione **imgGomma_MouseDown** gestisce l'evento MouseDown del bottone **Gomma** attivando sempre l'**inkCanvas** ma modificando il tratto in modalità cancellazione.

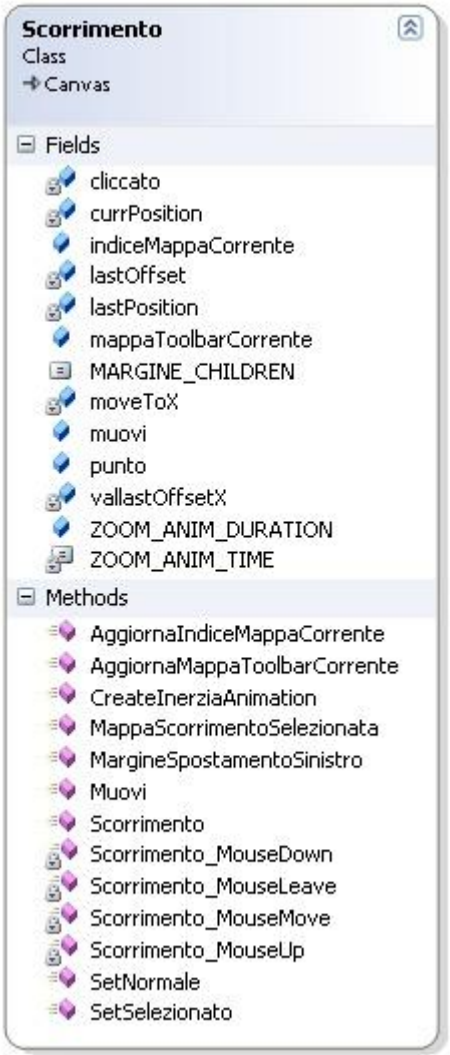
La funzione **imgZoom_MouseDown** gestisce l'evento MouseDown del bottone **Lente** visualizzando il canvas **muoviCanvas** che viene gestito nella classe **Mappa**.

La funzione **imgPunti_MouseDown** gestisce l'evento MouseDown del bottone **Punti Interattivi** visualizzando il canvas **Punti** dove appaiono dei punti interattivi nelle coordinate prestabilite e ad ognuno dei quali sono allegati i video e i documenti.

Il metodo **imgLibro_MouseDown** gestisce l'evento MouseDown del bottone **Libro** visualizzando il **BookControl** dove viene attivato lo sfogliamento delle pagine del libro. Inoltre se il bottone viene mantenuto premuto per almeno tre secondi si attiva il canvas **Scorrimento** dove sono caricate le immagini di media dimensione ed è possibile scorrere tutte le mappe dello specifico continente.

Immagine N° 6 – Punti interattivi

10.7 Classe Scorrimento



La classe **Scorrimento** è responsabile dello scorrimento orizzontale con visualizzazione delle immagini a media dimensione.

L'evento **Scorrimento_MouseMove** gestisce il movimento orizzontale del contenitore controllando che esso non esca dai bordi destro e sinistro.

L'evento **Scorrimento_MouseUp** gestisce l'inerzia sempre controllando i bordi destro e sinistro. Il movimento verso sinistra avviene selezionando e posizionando la successiva immagine al centro dell'area di lavoro, invece verso destra selezionando e posizionando l'immagine precedente al centro. Quando avviene lo spostamento del canvas vengono aggiornate la mappa centrale dell'area di lavoro, la selezione del menu scorrevole verticale a sinistra e la pagina corrente del libro.

Immagine N° 7 – Visualizzazione del menu di scorrimento orizzontale

11. Funzionalità

11.1 Videata principale

All’avvio del programma viene visualizzata la finestra principale di lavoro con gli strumenti che si trovano a destra mentre, nella parte inferiore dello schermo, c’è una **toolbar** con le **icone delle mappe** dei vari continenti.

Immagine N° 8 – Videata principale con la toolbar dei continenti

11.2 Toolbar

Immagine N° 9 – Toolbar dei continenti

La toolbar è un Canvas dove sono caricate le immagini principali di tutti continenti.

E' possibile muovere in orizzontale la toolbar con i continenti grazie all'aggiornamento della trasformata TranslateTransform associata. Il movimento avviene quando viene intercettato l'evento MouseMove calcolando semplicemente la differenza delle coordinate X dei punti dell'ultima posizione e della posizione corrente. Inoltre è stato gestito che la toolbar non esca dai bordi destro e sinistro dello schermo:

```
/// <summary>
/// Funzione che gestisce l'evento MouseMove sul canvas toolbar, gestione dei bordi
/// destro e sinistro
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void cnvToolbarMappe_MouseMove(object sender, MouseEventArgs e)
{
    if (cliccato)
    {
        // Prendo la posizione corrente
        currPosition = e.GetPosition(null);

        // Calcolo l'offset di spostamento
        lastOffset.X = currPosition.X - lastPosition.X;

        // Sposto l'immagine controllando i bordi sinistro e destro
        if (muovi.X + lastOffset.X < Globali.screenWidth - Globali.TOOLBAR_MARGINE_LEFT
          * 4 && muovi.X + lastOffset.X > (Globali.toolbar.Width -
            Globali.MARGINE_LEFT * 4))
        {
            // Sposto l'immagine
            muovi.X += lastOffset.X;

            // Prendo l'ultima posizione
            lastPosition = currPosition;
        }
    }
}
```

Il movimento con l'inerzia è stato realizzato grazie all'utilizzo di un'animazione DoubleAnimation. Per ottenere il movimento della toolbar con l'inerzia, al verificarsi dell'evento MouseUp viene ottenuto il punto di arrivo calcolando la velocità in base alla differenza tra le coordinate orizzontali del punto dell'ultimo tocco e il punto di distacco del dito. Dopodiché viene lanciata l'animazione chiamando la funzione di animazione inerzia **CreateInerziaAnimation**. Anche in questo caso è stato gestito che la toolbar non esca dai bordi dello schermo.

```
/// <summary>
/// Funzione che gestisce l'evento MouseUp sulla toolbar, gestione inerzia e gestione
/// bordi destro e sinistro
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void cnvToolbarMappe_MouseUp(object sender, MouseButtonEventArgs e)
{
    e.Handled = true;

    vallastOffsetX = Math.Abs(lastOffset.X);

    // Setto il flag di non cliccato
    cliccato = false;

    if (vallastOffsetX != 0)
    {
        // Calcolo la velocità
        if (vallastOffsetX > 5)
            Globali.coeffVelocitaX = Globali.VELOCITA_3;
        else if (vallastOffsetX > 3)
            Globali.coeffVelocitaX = Globali.VELOCITA_2;
        else if (vallastOffsetX > 2)
            Globali.coeffVelocitaX = Globali.VELOCITA_1;
        else if (vallastOffsetX >= 1)
            Globali.coeffVelocitaX = Globali.VELOCITA_0;
        else
            Globali.coeffVelocitaX = 0;

        // Calcolo il nuovo punto di arrivo dell'animazione inerzia
        moveToX = lastOffset.X * Globali.coeffVelocitaX;
        punto.X = (muovi.X + moveToX);

        #region gestione del limite sinistro e destro
        // Gestione del bordo destro
        if (punto.X > Globali.screenWidth - Globali.TOOLBAR_MARGINE_LEFT * 4)
            punto.X = Globali.screenWidth - Globali.toolbar.Width / 2;

        // Gestione del bordo sinistro
        if (punto.X < -(Globali.toolbar.Width - Globali.MARGINE_LEFT * 4))
            punto.X = -(3 * Globali.toolbar.Width / 4 - Globali.MARGINE_LEFT * 4);
        #endregion

        // Lancio l'animazione Inerzia orizzontale
        muovi.BeginAnimation(TranslateTransform.XProperty,
            CreateInerziaAnimation(punto.X));
    }
    lastOffset.X = 0;
}
```

Per **selezionare** un continente è possibile trascinare la relativa icona in verticale al centro dello schermo.

Immagine N° 10 – Selezione di un continente dalla toolbar

Il movimento dell'icona in verticale avviene grazie alla trasformata TranslateTransform, intercettando l'evento MouseMove. Il calcolo dello spostamento dell'icona verso l'alto è lo stesso di quello per il movimento della toolbar, ma prende in considerazione la coordinata Y.

Intercettando l'evento MouseUp al rilascio dell'icona, se la posizione corrente supera una certa distanza dal punto originario viene selezionato quel continente. Viene caricata la mappa principale del continente nelle sue dimensioni originali nell'area di lavoro, vengono caricati inoltre tutti gli oggetti necessari agli strumenti e relativi alla mappa centrale selezionata quali l'oggetto di tipo InkCanvas per poter disegnare/cancellare sulla mappa, l'immagine dell'effetto lente e il canvas Punti con i punti interattivi posizionati sulla mappa centrale.

Vengono caricate tutte le mappe del continente prescelto nel menu scorrevole verticale e, grazie alla lista di tipo MappaCategoria creata in fase di avvio dell'applicazione, viene evidenziata all'utente l'avvenuta selezione della mappa operando un cambiamento di colore del bordo sottostante da trasparente a rosso. Inoltre vengono caricate tutte le mappe del continente prescelto nello scorrimento orizzontale con immagini a dimensione media oltre che le pagine del libro sempre attingendo dalla lista di classi di tipo MappaCategoria.

Una volta scelto il continente, al fine di aumentare l'area di lavoro principale, la toolbar sparisce dallo schermo utilizzando un'animazione di tipo DoubleAnimation.

Immagine N° 11 – Continente selezionato

11.3 Menu del continente

Analogamente a quanto avviene per la toolbar dei continenti, è possibile **muovere** l'insieme delle icone del menu scorrevole in verticale. Il movimento avviene grazie all'azione sulla trasformata TranslateTransform collegata, intercettando l'evento MouseMove. Il calcolo dello spostamente è lo stesso della toolbar però opera sulla coordinata Y.

Il menu si muove in verticale con l'inerzia grazie ad un'animazione di tipo DoubleAnimation. Al verificarsi dell'evento MouseUp viene ottenuto il nuovo punto di arrivo calcolando la velocità in base alla differenza tra le coordinate verticali del punto dell'ultimo tocco e il punto di distacco del dito. Alla fine viene lanciata l'animazione chiamando il metodo **CreateInerziaAnimation**. Anche qui è stato gestito che il menu non fuoriesca dello schermo.

Immagine N° 12 – Menu scorrevole verticale

```
/// <summary>
/// Funzione che gestisce l'evento MouseUp sul canvas menu,
/// spostamento del menu in verticale, gestione inerzia e
/// gestione dei bordi superiore e inferiore
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void cnvMenuMappe_MouseUp(object sender, MouseButtonEventArgs e)
{
    vallastOffsetY = Math.Abs(lastOffset.Y);

    // Setto il flag di non cliccato
    cliccatoMenu = false;

    if (vallastOffsetY != 0)
    {
        // Calcolo la velocità
        if (vallastOffsetY > 5)
            Globali.coeffVelocitaY = Globali.VELOCITA_3;
        else if (vallastOffsetY > 3)
            Globali.coeffVelocitaY = Globali.VELOCITA_2;
        else if (vallastOffsetY > 2)
            Globali.coeffVelocitaY = Globali.VELOCITA_1;
        else if (vallastOffsetY >= 1)
            Globali.coeffVelocitaY = Globali.VELOCITA_0;
        else
            Globali.coeffVelocitaY = 0;

        // Calcolo il nuovo punto di arrivo dell'animazione
        // inerzia
        moveToY = lastOffset.Y * Globali.coeffVelocitaY;
        punto.Y = (Globali.muoviMenu.Y + moveToY);

        #region gestione del limite top e bottom
        // Gestione del bordo inferiore
        if (punto.Y > Globali.screenHeight -
          Globali.MARGINE_TOP_MENU * 4)
            punto.Y = Globali.screenHeight -Globali.menu.Height/2;

        // Gestione del bordo superiore
        if (punto.Y < -(Globali.menu.Height -
          Globali.MARGINE_TOP_MENU * 4))
            punto.Y = Globali.MARGINE_TOP_MENU -
              Globali.menu.Height / 2;
        #endregion

        // Lancio l'animazione Inerzia verticale
        Globali.muoviMenu.BeginAnimation(TranslateTransform.YProp
            erty, CreateInerziaAnimation(punto.Y));
    }

    lastOffset.Y = 0;
}
```

Immagine N° 13 – Menu scorrevole verticale

Per selezionare la mappa desiderata basta trascinarla in orizzontale al centro dello schermo e la stessa prende la dimensione massima dell'area di lavoro. L'immagine selezionata del menu avrà il bordo rosso.

Immagine N° 14 – Selezione mappa dal menu scorrevole verticale

Il movimento dell'icona in orizzontale avviene operando sulla trasformata TranslateTransform collegata all'evento MouseMove, come in precedenza si calcola la differenza delle coordinate X dello spostamento.

Quando viene intercettato l'evento MouseUp, se lo spostamento è maggiore di una certa distanza dal punto originale, avviene la selezione della mappa prescelta. Nella mappa centrale viene caricata l'immagine di dimensione originale, l'oggetto di tipo InkCanvas che serve per disegnare/cancellare sulla mappa, l'immagine dell'effetto lente e il canvas Punti con i punti interattivi collegati alla mappa prescelta.

L’icona selezionata del menu scorrevole cambia il colore del bordo da trasparente a rosso. Inoltre, anche se rimangono invisibili, vegono aggiornati lo scorrimento con le mappe a media dimensione e la pagina selezionata del libro.

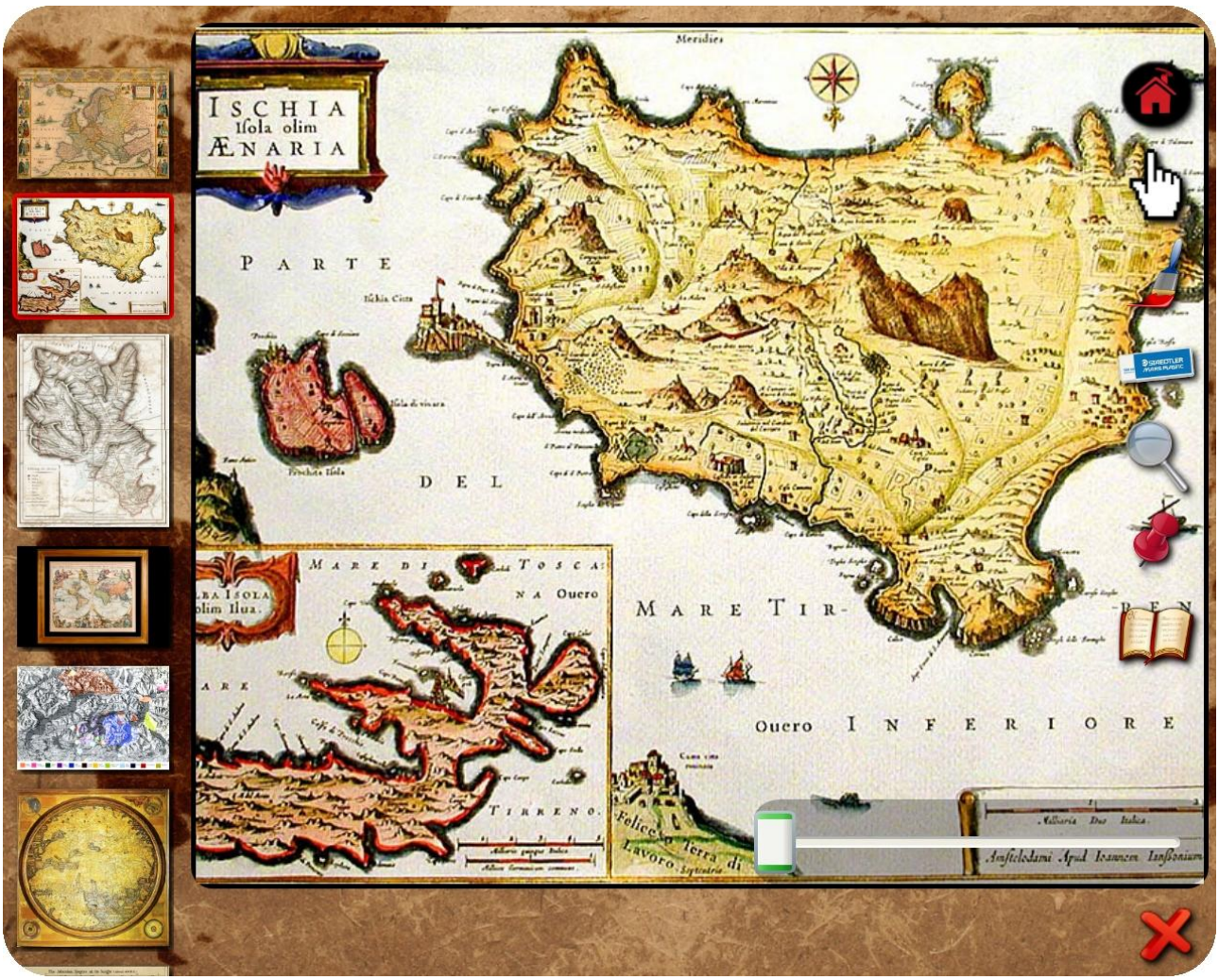


Immagine N° 15 – Mappa selezionata

11.4 Strumenti

Nella parte destra dell'area di lavoro ci sono gli strumenti/bottoni collegati alle relative funzioni. Il contenitore è un oggetto di tipo Canvas che contiene le immagini degli strumenti. I bottoni hanno l'effetto ombra. Quando viene toccato uno dei bottoni l'ombra dell'immagine diminuisce e il bottone viene spostato leggermente creando un effetto come fosse schiacciato:

```
/// <summary>
/// Funzione che setta il flag schiacciato del bottone
/// </summary>
public void SetSchiacciato()
{
    if (!schiacciato)
    {
        // Spostamento del bottone
        Canvas.SetLeft(this, Canvas.GetLeft(this) + OFFSET_SCHIACCIATO);
        Canvas.SetTop(this, Canvas.GetTop(this) + OFFSET_SCHIACCIATO);
    }

    schiacciato = true;
}
```

Bottone lente selezionato

Immagine N° 16 – Strumento selezionato

11.4.1 Home

Immagine N° 17 – Strumento home

Toccando il bottone **Home** viene riattivata, tramite un'animazione verticale, la **toolbar** con i continenti dando la possibilità di scegliere un altro continente.

Una volta scelta un'altra icona dalla toolbar, quest'ultima sparisce nuovamente e viene ingrandita l'area di lavoro.

11.4.2 Mano

Selezionando il bottone **Mano** è possibile muovere la mappa, inoltre con due dita è possibile ridimensionare l'immagine.

11.4.2.1 Movimento della mappa centrale

Immagine N° 18 – Strumento mano: spostamento della mappa selezionata

Per poter muovere la mappa centrale nella classe **Mappa**, che estende da un oggetto Image, è stato creato un contenitore di tipo Canvas denominato **muoviCanvas** dove viene caricata l'immagine di dimensione originale.

Il movimento avviene intercettando l'evento MouseMove e operando sulla trasformata TranslateTransform. Lo spostamento viene calcolato, analogamente allo

spostamento della toolbar con i continenti e a quello del menu scorrevole verticale, come differenza delle coordinate X e coordinate Y tra l'ultima posizione e quella corrente. E' stato gestito che la mappa non esca dai bordi dell'area di lavoro destro, sinistro, superiore e inferiore.

```
/// <summary>
/// Movimento della mappa centrale, gestione dei bordi destro, sinistro, superiore e
/// inferiore
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void Mappa_MouseMove(object sender, System.Windows.Input.MouseEventArgs e)
{
    if (cliccato)
    {
        // Prendo la posizione corrente
        currPosition = e.GetPosition(null);

        // Gestione del Movimento se cliccato il bottone muovi
        if (Globali.bottoni.GetBottoneCliccato() == CnvBottoniDisegno.BOTTONE_MUOVI)
        {
            // Metto offset a zero
            lastOffset.X = lastOffset.Y = 0;

            // Sposto l'immagine controllando i bordi sinistro e destro
            if ((currPosition.X - lastPosition.X > 1 && (muovi.X + (currPosition.X -
              lastPosition.X)) / scaleTransform.ScaleX < (Globali.screenWidth -
                Globali.MARGINE_LEFT * 20) / scaleTransform.ScaleX) || (currPosition.X -
                  lastPosition.X < -1 && (muovi.X + (currPosition.X - lastPosition.X)) /
                    scaleTransform.ScaleX > -(3 * imgMappa.Width / 4 -
                      Globali.MARGINE_LEFT) / scaleTransform.ScaleX))
            {
                // Calcolo l'offset X
                lastOffset.X = currPosition.X - lastPosition.X;

                // Aggiorno la trasformata di movimento relativamente alla coordinata X
                muovi.X += lastOffset.X / scaleTransform.ScaleX;
            }

            // Sposto l'immagine controllando i bordi superiore e inferiore
            if ((currPosition.Y - lastPosition.Y > 0 && (muovi.Y + (currPosition.Y -
              lastPosition.Y)) / scaleTransform.ScaleY < (Globali.MAPPA_HEIGHT_NUOVO -
                Globali.MARGINE_TOP * 5) / scaleTransform.ScaleY) || (currPosition.Y -
                  lastPosition.Y < 0 &&(muovi.Y + (currPosition.Y - lastPosition.Y)) /
                    scaleTransform.ScaleY > -(imgMappa.Height - Globali.MARGINE_TOP * 5)
                      / scaleTransform.ScaleY))
            {
                // Calcolo l'offset Y
                lastOffset.Y = currPosition.Y - lastPosition.Y;

                // Aggiorno la trasformata di movimento relativamente alla coordinata Y
                muovi.Y += lastOffset.Y / scaleTransform.ScaleY;
            }
        }

        // Aggiorno l'ultima posizione a quella corrente
        lastPosition = currPosition;
    }
}
```

Quando si verifica l'evento MouseUp l'immagine si muove con l'inerzia che è stata gestita analogamente all'inerzia della toolbar con i continenti e a quella del menu scorrevole verticale, cioè utilizzando un'animazione DoubleAnimation. Il calcolo è analogo a quello utilizzato per toolbar e menu verticale. Inoltre è stato gestito come sempre che l'immagine non esca dai bordi dell'area di lavoro.

```
/// <summary>
/// Funzione che gestisce l'evento MouseUp della mappa centrale, movimento della mappa
/// con inerzia, gestione dei bordi destro, sinistro, superiore e inferiore
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void Mappa_MouseUp(object sender, System.Windows.Input.MouseButtonEventArgs e)
{
    e.Handled = true;

    // Setto il flag di non cliccato
    cliccato = false;

    // Limite massimo di lanciamento
    if (Math.Abs(lastOffset.X) > 50)
        lastOffset.X = 50 * Math.Sign(lastOffset.X);

    vallastOffsetX = Math.Abs(lastOffset.X);
    vallastOffsetY = Math.Abs(lastOffset.Y);

    if ((vallastOffsetX != 0 || vallastOffsetY != 0) && scaleTransform.ScaleX < 5)
    {
        // Calcolo delle velocità
        // Velocità in orizzontale
        if (scaleTransform.ScaleX > 2)
            Globali.coeffVelocitaX = VELOCITA_0;
        else if (vallastOffsetX > 5)
            Globali.coeffVelocitaX = VELOCITA_3;
        else if (vallastOffsetX > 3)
            Globali.coeffVelocitaX = VELOCITA_2;
        else if (vallastOffsetX > 2)
            Globali.coeffVelocitaX = VELOCITA_1;
        else if (vallastOffsetX >= 1)
            Globali.coeffVelocitaX = VELOCITA_0;
        else
            Globali.coeffVelocitaX = 0;

        // Velocità in verticale
        if (scaleTransform.ScaleY > 2)
            Globali.coeffVelocitaY = VELOCITA_0;
        else if (vallastOffsetY > 5)
            Globali.coeffVelocitaY = VELOCITA_3;
        else if (vallastOffsetY > 3)
            Globali.coeffVelocitaY = VELOCITA_2;
        else if (vallastOffsetY > 2)
            Globali.coeffVelocitaY = VELOCITA_1;
        else if (vallastOffsetY >= 1)
            Globali.coeffVelocitaY = VELOCITA_0;
        else
            Globali.coeffVelocitaY = 0;
```

```
// Calcolo il nuovo punto di arrivo dell'animazione inerzia
moveToX = (lastOffset.X * Globali.coeffVelocitaX) / scaleTransform.ScaleX;
moveToY = (lastOffset.Y * Globali.coeffVelocitaY) / scaleTransform.ScaleY;
punto = new Point(muovi.X + moveToX, muovi.Y + moveToY);

#region Gestione del limite del bordo
// Gestione dei bordi se la mappa non è ingrandita
if (scaleTransform.ScaleX == 1)
{
    // Gestione del bordo destro
    if (lastOffset.X > 0 && punto.X >= (Globali.screenWidth -
      Globali.MARGINE_LEFT * 20) / scaleTransform.ScaleX)
        punto.X = (Globali.screenWidth - Globali.MARGINE_LEFT * 40) /
          scaleTransform.ScaleX;

    // Gestione del bordo sinistro
    if (lastOffset.X < 0 && punto.X <= -(3 * imgMappa.Width / 4 -
      Globali.MARGINE_LEFT) / scaleTransform.ScaleX)
        punto.X = -(imgMappa.Width / 2 - Globali.MARGINE_LEFT) /
          scaleTransform.ScaleX;

    // Gestione del bordo inferiore
    if (punto.Y >= (Globali.MAPPA_HEIGHT_NUOVO - Globali.MARGINE_TOP * 5) /
      scaleTransform.ScaleY)
        punto.Y = (Globali.MAPPA_HEIGHT_NUOVO - Globali.MARGINE_TOP * 10) /
          scaleTransform.ScaleY;

    // Gestione del bordo superiore
    if (punto.Y <= -(imgMappa.Height - Globali.MARGINE_TOP * 5) /
      scaleTransform.ScaleY)
        punto.Y = -(imgMappa.Height - Globali.MARGINE_TOP * 10) /
          scaleTransform.ScaleY;
}
// Gestione dei bordi se la mappa è ingrandita
else
{
    // Gestione del bordo destro
    if (lastOffset.X > 0 && punto.X >= (Globali.screenWidth -
      Globali.MARGINE_LEFT * 20) / scaleTransform.ScaleX * 4)
        punto.X = (Globali.screenWidth - Globali.MARGINE_LEFT * 40) /
          scaleTransform.ScaleX * 4;

    // Gestione del bordo sinistro
    if (lastOffset.X < 0 && punto.X <= -(3 * imgMappa.Width / 4 -
      Globali.MARGINE_LEFT) / scaleTransform.ScaleX * 4)
        punto.X = -(imgMappa.Width / 2 - Globali.MARGINE_LEFT) /
          scaleTransform.ScaleX * 4;

    // Gestione del bordo inferiore
    if (punto.Y >= (Globali.MAPPA_HEIGHT_NUOVO - Globali.MARGINE_TOP * 5) /
      scaleTransform.ScaleY * 4)
        punto.Y = (Globali.MAPPA_HEIGHT_NUOVO - Globali.MARGINE_TOP * 10) /
          scaleTransform.ScaleY * 4;

    // Gestione del bordo superiore
    if (punto.Y <= -(imgMappa.Height - Globali.MARGINE_TOP * 5) /
      scaleTransform.ScaleY * 4)
        punto.Y = -(imgMappa.Height - Globali.MARGINE_TOP * 10) /
          scaleTransform.ScaleY * 4;
}
#endregion
```

```
        // Lancio l'animazione Inerzia
        muovi.BeginAnimation(TranslateTransform.XProperty,
            CreateInerziaAnimation(punto.X));

        muovi.BeginAnimation(TranslateTransform.YProperty,
            CreateInerziaAnimation(punto.Y));
    }
}
```

11.4.2.2 Multitouch

Per la gestione multitouch, che permette di zoomare l'immagine con due dita, è stata importata la libreria **NWMultiTouch.dll** fornita dal produttore (NextWindow) della tecnologia touchscreen che gestisce l'interfacciamento con l'hardware installato sulla stazione HP TouchSmart 512it.

Vengono contati i tocchi contemporanei sullo schermo e se vengono verificati due tocchi sullo stesso oggetto cioè sulla mappa centrale, utilizzando il teorema di Pitagora viene calcolato il coefficiente di zoom con cui aggiornare l'immagine.

Viene infine aggiornato lo slider dello zoom che è collegato alla trasformata di ridimensionamento, di tipo ScaleTransform, dell'immagine correntemente selezionata. Al verificarsi invece di un solo tocco viene spostata la mappa agendo sulla trasformata di movimento associata di tipo TranslateTransform.

Funzione di gestione dell'evento multitouch:

```
/// <summary>
/// Funzione che gestisce il multitouch e il ridimensionamento della mappa
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void DrawTimer_Tick(object sender, EventArgs e)
{
    if (Globali.bottoni.GetBottoneCliccato() == CnvBottoniDisegno.BOTTONE_MUOVI)
    {
        …
        // Se è stato selezionato un contenuto scala
        if (contentHitted)
        {
            // Scaling – Teorema di Pitagora
            currentLength = Math.Sqrt(Math.Pow((tPts[1].x - tPts[0].x), 2) +
                Math.Pow((tPts[1].y - tPts[0].y), 2));


            if (lastLength != 0)
            {
                deltaScale = currentLength / lastLength;
```

```
                // Aggiorno lo slider dello zoom
                Globali.bordoZoom.AggiornaSlider(deltaScale);
            }
            lastPosition = tPts[0];
            lastLength = currentLength;
            lastpt1 = new System.Windows.Point(tPts[0].x, tPts[0].y);
            lastpt2 = new System.Windows.Point(tPts[1].x, tPts[1].y)
        }
        …
    }
    …
}
```

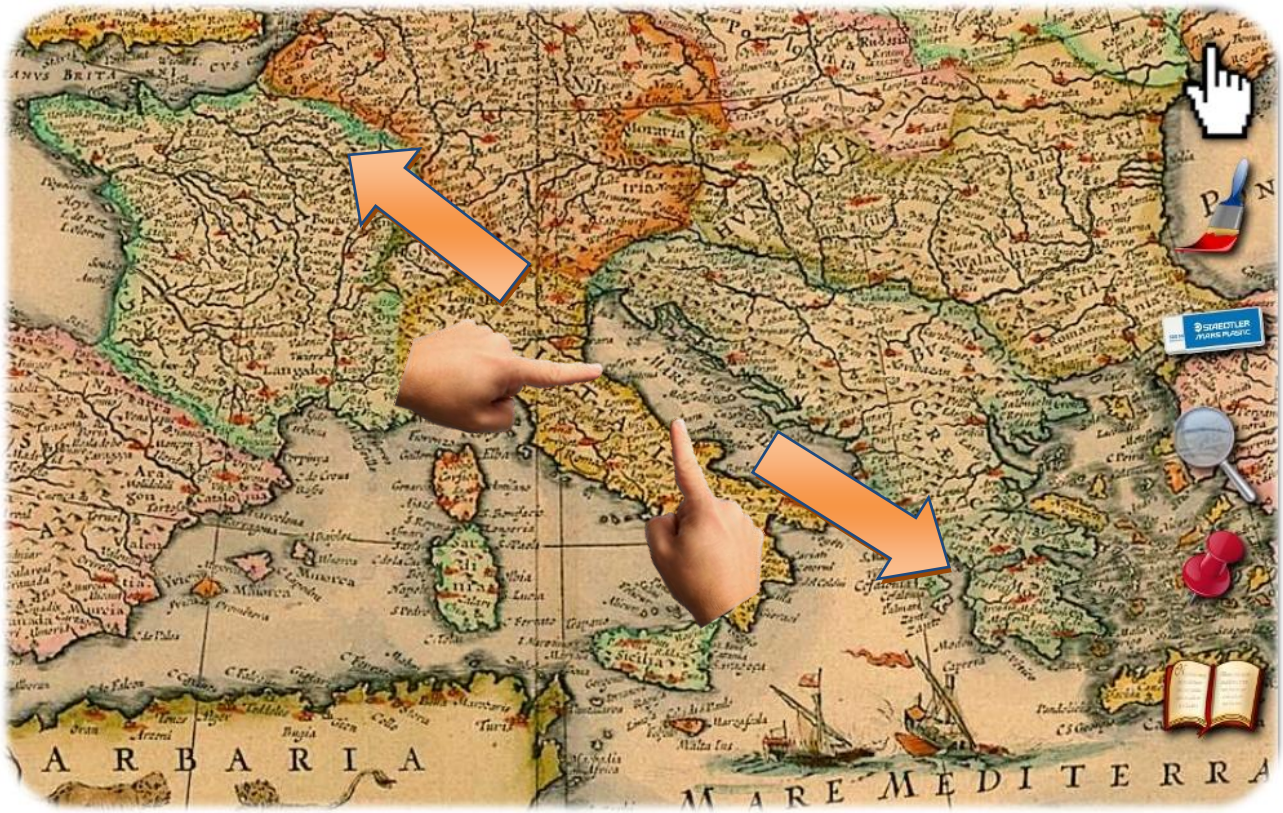


Immagine N° 19 – Strumento mano: ridimensionamento multitouch

11.4.3 Pennello

11.4.3.1 Disegnare con un dito

La selezione del bottone **Pennello** dà la possibilità di scrivere o disegnare sull'immagine centrale.

Immagine N° 20 – Strumento pennello: disegno su mappa

Per poter disegnare sull'immagine centrale è stato aggiunto un oggetto di tipo InkCanvas delle stesse dimensioni dell'immagine che permette di aggiungere delle scritte o dei disegni.

Di default è stato scelto il colore del pennello rosso:

```
// Setto il colore iniziale del pennello a rosso
attributiPennello.Color = Colors.Red;

// Definisco lo spessore del pennello
attributiPennello.Width = attributiPennello.Height = 3;
```

Selezionando il bottone **Pennello** l'oggetto di tipo InkCanvas diventa visibile e viene assegnata la proprietà InkCanvasEditingMode.Ink che permette di aggiungere scritte o disegni:

```
Globali.inkCentrale.EditingMode = InkCanvasEditingMode.Ink;
Globali.inkCentrale.Visibility = Visibility.Visible;
```

11.4.3.2 Cambiare il colore del pennello

Per poter cambiare il colore del **Pennello** basta tener premuto il relativo bottone per almeno tre secondi, dopodiché appare la finestra "**Tavola Colori**" che permette la selezione.

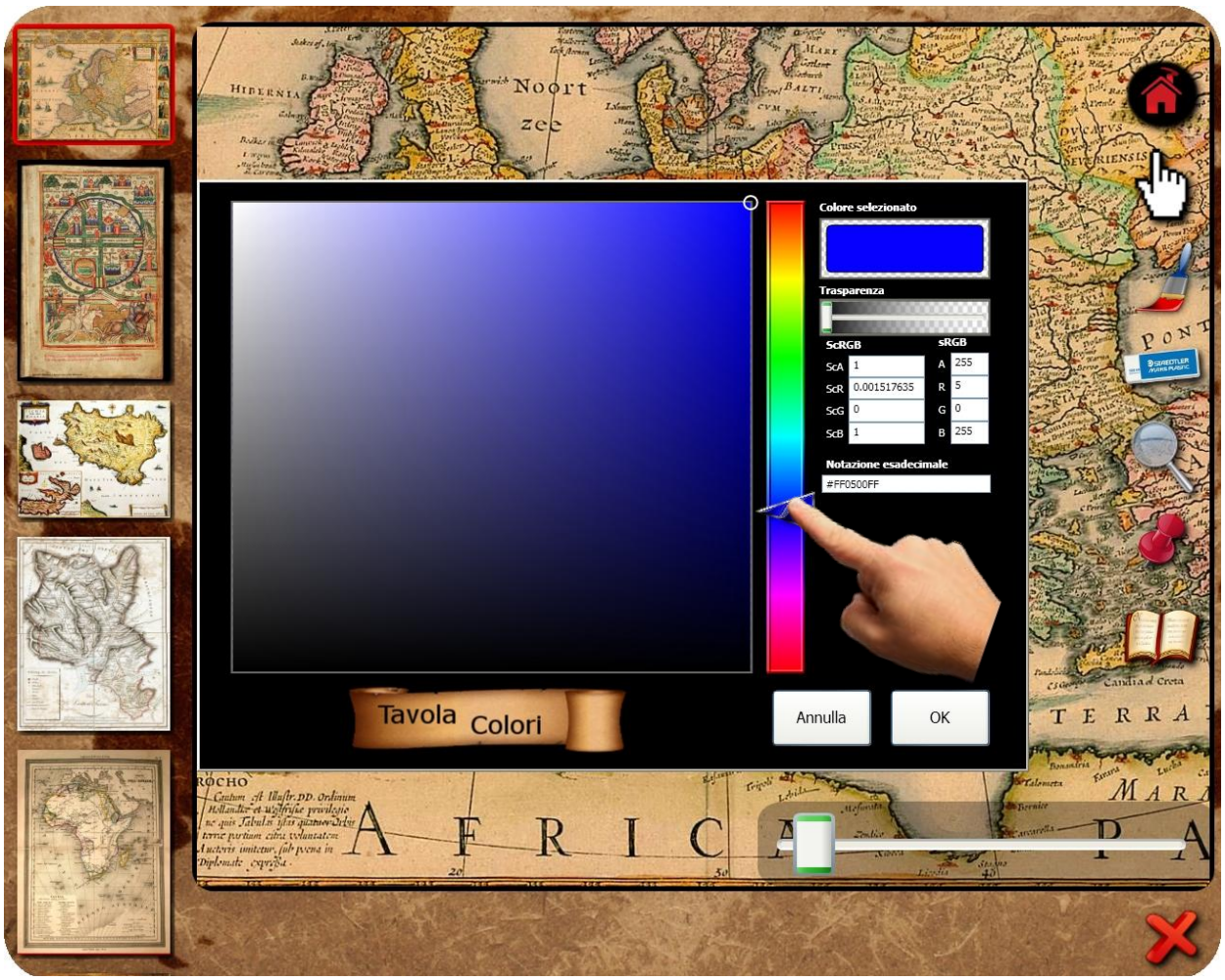


Immagine N° 21 – Strumento pennello: selezione del colore

Per poter cambiare il colore del **Pennello** è stata importata una libreria contenente un singolo oggetto che gestisce la selezione del colore:

```
try
{
    // Creo l'istanza dell'oggetto di selezione colore
    tavolaColori = new Microsoft.Samples.CustomControls.ColorPickerDialog();
    tavolaColori.WindowStartupLocation = WindowStartupLocation.CenterScreen;

    // Inizializzo il colore iniziale con quello selezionato precedentemente
    tavolaColori.StartingColor = Globali.attributiPennello.Color;

    // Se l'utente conferma la scelta, aggiorno il colore selezionato del pennello
    if ((bool)tavolaColori.ShowDialog())
    {
        Globali.attributiPennello.Color = tavolaColori.SelectedColor;
        Globali.inkCentrale.DefaultDrawingAttributes.Color =
            Globali.attributiPennello.Color;
    }
}
catch
{}
```

Immagine N° 22 – Strumento pennello: disegno con colori diversi

11.4.4 Gomma

Selezionando il bottone **Gomma** è possibile cancellare le scritte o i disegni fatti con il pennello.

Immagine N° 23 – Strumento gomma: cancellazione disegni

Analogamente alla selezione del bottone Pennello, l'oggetto di tipo InkCanvas diventa visibile e viene assegnata la proprietà che dà la possibilità di cancellare i disegni fatti precedentemente con il Pennello:

```
Globali.inkCentrale.EditingMode = InkCanvasEditingMode.EraseByPoint;
Globali.inkCentrale.Visibility = Visibility.Visible;
```

11.4.5 Effetto lente

La selezione del bottone **Lente** dà la possibilità di ingrandire l'area toccata producendo un effetto "**lente d'ingrandimento**".

Immagine N° 24 – Strumento lente: effetto lente d'ingrandimento su un particolare della mappa

Nella classe **Mappa** è stata creata un'immagine clonata dell'immagine originale che, grazie alla trasformata ScaleTransform, viene ridimensionata di un coefficiente prestabilito intorno alla posizione del tocco sull'immagine originale.

Al verificarsi dell'evento MouseDown viene aggiornato il punto toccato chiamando la funzione **UpdateLente** e l'immagine viene ingrandita di un raggio prestabilito intorno al punto. Dopodiché viene lanciata l'animazione operante sulla proprietà **Opacity** che fa apparire il particolare ingrandito.

Evento MouseDown:

```
// Aggiorno la lente con le coordinate di contatto del dito con lo schermo
UpdateLente(e.GetPosition(imgMappa));

// Animazione per far apparire la lente
imgEffect.BeginAnimation(Image.OpacityProperty, mAnim);
```

Funzione di aggiornamento della posizione della Lente:

```
/// <summary>
/// Funzione che aggiorna il centro della lente
/// </summary>
/// <param name="position">Il punto toccato</param>
void UpdateLente(Point position)
{
    if (scaleTransform.ScaleX == 1)
        geometria.RadiusX = geometria.RadiusY = LENTE_SIZE;
    else
        geometria.RadiusX = geometria.RadiusY = LENTE_SIZE / (scaleTransform.ScaleX *
            LENTE_ZOOM_FACTOR);

    // Aggiorno la posizione del centro della lente
    geometria.Center = position;

    // Sposto l'immagine al punto toccato
    imgEffect.SetValue(Canvas.LeftProperty, -(position.X) + (this.Width –
        imgMappa.Width) / 2);
    imgEffect.SetValue(Canvas.TopProperty, -(position.Y));
}
```

L'aggiornamento del punto toccato avviene anche al verificarsi dell'evento MouseMove.

Quando viene intercettato l'evento MouseUp viene lanciata l'animazione operante sulla proprietà **Opacity** che fa sparire l'effetto lente.

```
// Animazione per far sparire la lente
imgEffect.BeginAnimation(Image.OpacityProperty, null);
imgEffect.BeginAnimation(Image.OpacityProperty, mBackAnim);
```

Immagine N° 25 – Strumento lente

11.4.6 Punti interattivi

Selezionando il bottone **Punti Interattivi** sulla mappa centrale appaiono le bandierine nei punti prestabiliti.

Immagine N° 26 – Strumento punti interattivi: attivazione punti interattivi degli eventi storici

All'attivazione del bottone **Punti Interattivi** diventa visibile l'oggetto **Punti** grazie all'animazione DoubleAnimation. La classe **Punti** estende dall'oggetto di tipo Canvas.

Questo canvas è, per la mappa selezionata, il contenitore di tutti punti interattivi con la descrizione dell'evento storico, dei video e dei documenti per ogni punto. I punti interattivi sono stati posizionati alle coordinate prestabilite (lette dal file di testo durante la fase di avvio).

La lettura del file di testo avviene riga per riga. Da ogni riga vengono ottenuti i dati per il punto con le coordinate X e Y, per la descrizione e per gli **n** percorsi dei video e **n** percorsi dei documenti.

Leggendo i dati, che sono divisi tra loro dal simbolo "**;**", per ogni punto viene creata una lista di tipo string dei percorsi video e una lista di tipo string dei percorsi documento, dove vengono aggiunte le informazioni lette. Dopodiché il punto, la descrizione dell'evento storico, la lista dei video e la lista dei documenti vengono aggiunti alla lista di classi **PuntiMappa** che è un oggetto che estende dalla classe StackPanel. Infine alla posizione della coordinata letta viene aggiunto lo StackPanel con l'immagine del punto interattivo.

Lettura del file txt:

```
StreamReader sr = File.OpenText(filePuntiMappa);
while ((tmp = sr.ReadLine()) != null)
{
    // Divido la stringa letta dal file in un vettore di stringhe
    string[] word = tmp.Split(';');

    // Prendo le coordinate del punto
    punto.X = Convert.ToDouble(word[0]);
    punto.Y = Convert.ToDouble(word[1]);

    // Creo la lista dei percorsi video
    List<string> lstVideo = new List<string>();

    // Creo la lista dei percorsi documento
    List<string> lstDocumento = new List<string>();

    // Aggiungo alla lista video il percorso del video e alla lista documenti il
    // percorso del documento
    for (int indice = 3; indice < word.Length; indice++)
    {
        if (word[indice].StartsWith("vid:"))
            lstVideo.Add(Globali.percorsoMappe + directory + "\\" +
                word[indice].Substring(4));
        else if (word[indice].StartsWith("doc:"))
            lstDocumento.Add(Globali.percorsoMappe + directory + "\\" +
                word[indice].Substring(4));
    }

    // Aggiungo alla lista i punti, la descrizione e per ogni punto le liste dei
    // percorsi video e dei percorsi documenti
    lstDescrizionePunti.Add(new PuntiMappa(punto, word[2], lstVideo, lstDocumento));

    // Aggiungo il punto al canvas contenitore dei punti
    this.Children.Add((PuntiMappa)lstDescrizionePunti.Last());
    // Posiziono il punto
    Canvas.SetLeft((PuntiMappa)lstDescrizionePunti.Last(), punto.X);
    Canvas.SetTop((PuntiMappa)lstDescrizionePunti.Last(), punto.Y);
}
sr.Close();
```

Toccando uno dei punti interattivi sulla mappa appaiono la descrizione dell'evento storico, le icone dei video e quelle dei documenti.

Immagine N° 27 – Strumento punti interattivi: visualizzazione evento storico con descrizione e file allegati

L'apparire della descrizione, delle icone dei video e dei documenti è gestita nella classe PuntiMappa che estende dalla classe StackPanel dove sono stati aggiunti in verticale un oggetto di tipo TextBlock per la descrizione, un oggetto di tipo StackPanel per i video e un oggetto di tipo StackPanel per i documenti.

Quindi all'intercettazione dell'evento MouseDown si lancia l'animazione che fa apparire la descrizione dell'evento storico, dei video e dei documenti.

```
/// <summary>
/// Funzione che Attiva/Disattiva la descrizione, bottoni video e bottoni documento
/// della mappa
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void puntoImage_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)
{
    // Animazione per far apparire la descrizione, i bottoni dei video e quelli dei
    // documenti
    if (brdDescrizione.Opacity == 0)
    {
        // Descrizione
        brdDescrizione.Visibility = Visibility.Visible;
        brdDescrizione.BeginAnimation(Canvas.OpacityProperty, Globali.opacityAnim);

        // Video
        stVideo.Visibility = Visibility.Visible;
        stVideo.BeginAnimation(Canvas.OpacityProperty, Globali.opacityAnim);

        // Documento
        stDocumento.Visibility = Visibility.Visible;
        stDocumento.BeginAnimation(Canvas.OpacityProperty, Globali.opacityAnim);
    }
    // Animazione per far sparire la descrizione, i bottoni dei video e quelli dei
    // documenti
    else
    {
        brdDescrizione.BeginAnimation(Canvas.OpacityProperty, opacityBackAnim);
        stVideo.BeginAnimation(Canvas.OpacityProperty, opacityBackAnim);
        stDocumento.BeginAnimation(Canvas.OpacityProperty, opacityBackAnim);
    }
    e.Handled = true;
}
```

Al verificarsi dell'evento MouseDown su uno dei bottoni collegati al video viene lanciato il programma esterno per esecuzione del video:

```
// Prendo il percorso del video
String percorsoFileVideo = imgVideo.Tag.ToString();

Process[] processi = Process.GetProcesses();
try
{
    // Se il processo è già attivo lo termino
    foreach (Process processo in processi)
    {
        if (processo.ProcessName.ToLower() == "vlc")
            processo.Kill();
    }

    // Lancio il programma di visualizzazione del video
    Process exeEsterno = new Process();
    exeEsterno.StartInfo.FileName = Globali.percorsoExeVideo;
    exeEsterno.StartInfo.Arguments = percorsoFileVideo;
    exeEsterno.Start();
}
```

```
catch (Exception ex)
{
    MessageBox.Show("Errore configurazione in config.ini del player video.");
}
```

Al verificarsi dell'evento MouseDown sul bottone documento viene lanciato il programma esterno per la visualizzazione del documento pdf:

```
// Prendo il percorso del documento
String percorsoFileDocumento = imgDocumento.Tag.ToString();

// Lancio il programma di visualizzazione del documento
try
{
    Process exeEsterno = new Process();
    exeEsterno.StartInfo.FileName = Globali.percorsoExeAdobe;
    exeEsterno.StartInfo.Arguments = percorsoFileDocumento;
    exeEsterno.Start();
}
catch (Exception ex)
{
    MessageBox.Show("Errore configurazione in config.ini del player Pdf.");
}
```

11.4.7 Libro

L'attivazione del bottone **Libro** fa apparire un oggetto **Book** di tipo **BookControl** il quale estende dall'oggetto UserControl.

Per la gestione del libro è stata aggiunta una libreria esterna che ho modificato e ricompilato.

Come è stato detto precedentemente, le immagini nel libro vengono caricate nel momento della scelta del continente. L'aggiornamento della pagina aperta avviene quando viene scelta l'immagine dal menu scorrevole verticale.

La selezione della pagina corrente viene evidenziata con un bordo rosso. All'apertura del libro è stato scelto di selezionare la pagina dispari. La selezione della pagina pari avviene tramite l'intercettazione dell'evento MouseDown.

Al verificarsi dell'evento **PageTurn**, che ho aggiunto alla libreria e dove viene gestita la pagina corrente, viene aggiornato il menu scorrevole verticale evidenziando l'icona selezionata con il bordo rosso, inoltre il menu scorrevole viene spostato muovendo l'icona selezionata in mezzo dello schermo.

Oltre al menu scorrevole verticale viene aggiornato anche lo scorrimento orizzontale e inoltre la mappa centrale dell'area di lavoro.

Immagine N° 28 – Strumento libro: sfogliamento delle pagine del libro

Gestione dell'evento di aggiornamento della pagine del libro:

```
/// <summary>
/// Evento PageTurn dove gestisco la pagina corrente
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void myBook_PageTurn(object sender, RoutedEventArgs e)
{
    // Disattivo il bordo dell'immagine selezionata precedentemente
    (Globali.Book.myBook.Items[Globali.cnvScorrimento.indiceMappaCorrente] as
        Border).BorderBrush = Brushes.Transparent;

    // Attivo il bordo dell'immagine selezionata
    Globali.cnvScorrimento.indiceMappaCorrente =Globali.Book.myBook.CurrentSheetIndex*2;

    // Se l'ultima pagina ha indice pari la seleziono altrimenti seleziono la dispari
    if (Globali.Book.myBook.Items.Count <= Globali.cnvScorrimento.indiceMappaCorrente)
        Globali.cnvScorrimento.indiceMappaCorrente--;

    (Globali.Book.myBook.Items[Globali.cnvScorrimento.indiceMappaCorrente] as
        Border).BorderBrush = Brushes.Red;

    // Aggiorno il menu con la mappa selezionata
    MappaToolbar.MappaMenuSelezionata(Globali.cnvScorrimento.indiceMappaCorrente);
}
```

11.4.8 Menu scorrevole orizzontale a media dimensione

Tenendo premuto il bottone **Libro** per almeno tre secondi appare il menu scorrevole orizzontale con le immagini a media dimensione. Il caricamento delle mappe avviene nel momento della selezione del continente. La selezione della mappa avviene nel momento della scelta della mappa dal menu oppure dal **Libro**.

La selezione della mappa consiste nel cambiamento del colore del bordo dell'immagine da trasparente a rosso e nello spostamento dell'oggetto di tipo Canvas, che è il contenitore delle immagini, in modo che la mappa selezionata si trovi al centro dell'area di lavoro.

Immagine N° 29 – Strumento menu scorrevole orizzontale: scorrimento delle mappe

Lo spostamento del Canvas di scorrimento avviene con l'inerzia analogamente alla toolbar dei continenti, è anche stato gestito che l'oggetto non esca dai bordi destro e sinistro dell'area di lavoro.

Scorrendo le mappe in orizzontale se lo spostamento è a sinistra, al verificarsi dell'evento MouseUp il **Canvas** di scorrimento si sposta in modo che la successiva mappa si trovi in mezzo dell'area di lavoro, viene cambiato il colore del bordo da trasparente a rosso, viene aggiornata la pagina dell'oggetto Book, viene selezionata l'icona successiva del menu scorrevole verticale, il medesimo menu si sposta mettendo l'immagine selezionata in mezzo allo schermo e infine viene aggiornata la mappa centrale. Se lo spostamento è a destra le stesse operazioni avvengono con l'immagine precedente.

Funzione di caricamento delle immagini al Canvas Scorrimento:

```
/// <summary>
/// Funzione che aggiunge le mappe al canvas scorrimento
/// </summary>
/// <param name="indice">Indice della mappa corrente</param>
public void AggiungiMappeSfogliamento(int indice)
{
    // Tolgo le eventuali mappe esistenti dal canvas scorrimento
    while (Globali.cnvScorrimento.Children.Count > 0)
        Globali.cnvScorrimento.Children.RemoveAt(0);

    widthPrecedente = 0;
    int i = 0;

    // Aggiungo le mappe al canvas Scorrimento che ritorna la width di ogni
    // immagine aggiunta
    while (i < lstMappeCategoria.Count)
        widthPrecedente += AggiungiMappaScorrimento(i++, widthPrecedente);

    // Ridimensiono il canvas scorrimento in base alla somma delle width di tutte le
    // mappe che ho aggiunto
    Globali.cnvScorrimento.Width = widthPrecedente;

    // Aggiorno lo Scorrimento con l'indice della mappa selezionata
    Globali.cnvScorrimento.AggiornaIndiceMappaCorrente(indice);
}
```

L'evento MouseUp dello scorrimento:

```
// Trovo l'immagine successiva
if (punto.X < -MargineSpostamentoSinistro(indiceMappaCorrente) -
    (this.Children[indiceMappaCorrente] as Border).Width / 2)
{
    // Se c'è ancora una mappa a destra seleziono quella
    if (indiceMappaCorrente < mappaToolbarCorrente.lstMappeCategoria.Count - 1)
    {
        Globali.UpdateCursor(Cursors.Wait);
```

```
        // Incremento indice
        indiceMappaCorrente++;

        // Seleziono l'immagine del menu
        MappaToolbar.MappaMenuSelezionata(indiceMappaCorrente);

        // Seleziono la pagina del book giusta
        Globali.Book.DeselezionaItems();
        Globali.Book.myBook.CurrentSheetIndex = indiceMappaCorrente % 2 == 0 ?
            indiceMappaCorrente / 2 : (indiceMappaCorrente + 1) / 2;

        // Aggiungo il bordo rosso alla pagina selezionata
        (Globali.Book.myBook.Items[indiceMappaCorrente] as Border).BorderBrush =
            Brushes.Red;

        // Sposto il canvas scorrimento a sinistra in modo che sia visibile la mappa
        // selezionata del menu
        TranslateTransform muoviSfogliamento = Globali.cnvScorrimento.RenderTransform as
            TranslateTransform;

        muoviSfogliamento.BeginAnimation(TranslateTransform.XProperty,
            MappaToolbar.CreateScorrimentoAnimationInit(-
                Globali.cnvScorrimento.MargineSpostamentoSinistro(indiceMappaCorrente)));

        // Seleziono la mappa centrale di scorrimento
        MappaScorrimentoSelezionata(indiceMappaCorrente);

        Globali.UpdateCursor(Cursors.Arrow);
    }
}
// Trovo l'immagine precedente
else if (punto.X > -MargineSpostamentoSinistro(indiceMappaCorrente) +
(this.Children[indiceMappaCorrente] as Border).Width / 2)
{
    // Se c'e' ancora una mappa a sinistra seleziono quella
    if (indiceMappaCorrente > 0)
    {
        Globali.UpdateCursor(Cursors.Wait);

        // Decremento l'indice
        indiceMappaCorrente--;

        // Seleziono l'immagine del menu
        MappaToolbar.MappaMenuSelezionata(indiceMappaCorrente);

        // Seleziono la pagina del book giusta
        Globali.Book.DeselezionaItems();
        Globali.Book.myBook.CurrentSheetIndex = indiceMappaCorrente % 2 == 0 ?
            indiceMappaCorrente / 2 : (indiceMappaCorrente + 1) / 2;

        // Aggiungo il bordo rosso alla pagina selezionata
        (Globali.Book.myBook.Items[indiceMappaCorrente] as Border).BorderBrush =
            Brushes.Red;

        // Sposto il canvas sfogliamento a sinistra in modo che sia visibile la mappa
        // selezionata del menu
        TranslateTransform muoviScorrimento = Globali.cnvScorrimento.RenderTransform as
            TranslateTransform;
```

```
            muoviScorrimento.BeginAnimation(TranslateTransform.XProperty,
                MappaToolbar.CreateScorrimentoAnimationInit(-
                    Globali.cnvScorrimento.MargineSpostamentoSinistro(indiceMappaCorrente)));

            // Seleziono la mappa centrale di scorrimento
            MappaScorrimentoSelezionata(indiceMappaCorrente);
        }
    }
```

Immagine N° 30 – Strumento menu scorrevole orizzontale: selezione della mappa

11.4.9 Slider per lo zoom

Lo slider dello zoom è gestito nella classe **BorderZoom** che estende dall'oggetto di tipo Border. L'oggetto denominato **zoomSlider** di tipo Slider è stato aggiunto nel contenitore BorderZoom.

I valori dello slider vanno da un minimo di 1 ad un massimo di 10.

Configurazione dello slider:

```
// Configuro i valori dello slider
zoomSlider.Maximum = 10;
zoomSlider.Minimum = 1;
zoomSlider.Value = 1.1;
```

Al verificarsi dell'evento **ValueChanged**, operando sulla trasformata ScaleTransform, la mappa centrale dell'area di lavoro viene ridimensionata.

Immagine N° 31 – Strumento slider zoom: ridimensionamento mappa selezionata

```
/// <summary>
/// Funzione che gestisce il ridimensionamento dell'immagine
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void zoomSlider_ValueChanged(object sender,
System.Windows.RoutedPropertyChangedEventArgs<double> e)
{
    if (Globali.mappaCentrale.Children.Count  > 0)
    {
        // Ottengo lo slider che è l'oggetto da cui arriva l'evento
        Slider slider = (Slider)sender;

        // Ottengo la trasformata di ridimensionamento della mappa selezionata
        ScaleTransform scale = Globali.mappaCentrale.Children[0].RenderTransform as
            ScaleTransform;

        // Aggiorno la dimensione operando sulla trasformata di ridimensionamento
        scale.ScaleX = scale.ScaleY = slider.Value;
    }
}
```

11.5 Exit

In basso a destra dello schermo è presente il bottone **Exit** attivando il quale si esce dall'applicazione:

```
/// <summary>
/// Gestione dell'evento MouseDown sul bottone Exit
/// </summary>
/// <param name="sender">Oggetto da cui arriva l'evento</param>
/// <param name="e">Argomento dell'evento</param>
void imgClose_MouseDown(object sender, MouseButtonEventArgs e)
{
    // Esco dall'applicazione
    Environment.Exit(0);
}
```

Immagine N° 32 – Bottone exit: bottone di uscita dal programma

12. Conclusioni

Lo scopo di questo progetto era di realizzare un'applicazione per sistemi touchscreen che permettesse una facile ed intuitiva navigazione dei contenuti (immagini, video, documenti, etc.) utilizzando solamente le dita. Quindi anche per gli utenti che non sanno utilizzare il computer.

Facendo un'analisi conclusiva del lavoro svolto, si può dire che l'obiettivo di progetto è stato raggiunto. Il lavoro ha portato alla realizzazione di un prototipo per la navigazione di mappe storiche digitalizzate (il contenuto potrebbe essere diverso: libri antichi, cataloghi, documenti, etc.).

Il programma applicativo ha diverse funzionalità come lo spostamento dei contenuti, il loro ridimensionamento, la possibiltà di disegnare/cancellare, l'ingrandimento della mappa nei punti desiderati, la consultazione di documenti e video collegati ai diversi punti interattivi e infine lo sfogliamento delle immagini come fossero le pagine di un libro.

La progettazione ha portato alla realizzazione di un applicativo costituito da **18** classi e circa **5000** righe di codice.

Nel corso della programmazione non sono emersi particolari problemi e la curva di apprendimento del linguaggio C# + WPF è risultata accettabile. Questo ha dimostrato che la soluzione tecnologica prescelta è stata adeguata alla realizzazione concreta del progetto.

Le difficoltà incontrate durante lo sviluppo sono state prevalentemente legate allo sfruttamento dell'hardware e in particolare della scheda video, alla realizzazione di effetti ritrovabili in natura come l'inerzia, alla realizzazione di movimenti intuitivi, alla fluidità delle animazioni e alla gestione e controllo dell'interazione multitouch.

Il giudizio personale sull'esperienza maturata durante la realizzazione dell'applicazione è sicuramente positivo. Grazie a questo progetto ho acquisito nuove conoscenze ed ho approfittato per aggiungere al mio bagaglio una nuova interessante tecnologia. Le varie problematiche sopraggiunte in fase di realizzazione hanno contribuito notevolmente alla mia formazione pratica e mi hanno permesso di conoscere le diverse fasi della produzione di un software applicativo.

13. Bibliografia

Christian Nagel, Bill Evjen, Jay Glynn, Morgan Skinner, Karli Watson
"C# 2008 Guida per lo sviluppatore"

Matthew MacDonald
"Pro WPF in C# 2010 Windows Presentation Foundation in .NET 4"

Sito della guida online di WPF in C#:

http://msdn.microsoft.com/it-it/library

Blog in lingua russa di WPF:

http://programmersforum.ru/forumdisplay.php?f=41

Forum in lingua russa di programmazione in C#:

http://www.cyberforum.ru/csharp-net/