BorrowBookControl

**void** setUI(IBorrowBookUI ui);

**Preconditions:** ui exists

Associates an instance of IBorrowBookUI with the control class.

Can only be called in the INITIALISED state.

Sets UI state to SWIPING

Transitions control class to SWIPING state.

**Postconditions:** UI and control class in SWIPING states

**void** cardSwiped(**int** patronId);

**Preconditions:** UI and control in SWIPING state, patronId is integer

Gets a Patron from library using patronId

Associates Patron with current instance of use case

If patronId not found

displays invalid patronID message to UI, exits

Checks with library whether patron has borrowing restrictions in place.

If patron can borrow,

Sets UI state to SCANNING

Transitions control class to SCANNING state.

If borrowing restricted,

displays restricted message to UI,

Sets UI state to RESTRICTED

Transitions control class to RESTRICTED state.

**Postconditions:**

patron not found – state remains SWIPING

patron is restricted from borrowing – control and UI in RESTRICTED state

patron can borrow – control and UI in SCANNING state, control has reference to patron

**void** bookScanned(**int** bookId);

**Preconditions:** UI and control in SCANNING state, control has reference to patron

Gets a Book from library using bookId

If bookId not found

displays invalid bookId message to UI, exits

checks with library if book is available

if book is not available

displays book not available message to UI, exits

gets library to create a pending loan, adds it to a pending loan list

for all current pending loans

displays book information for each loan to UI

checks with library if the patron will reach the maximum loans permitted with the current pending loan

displays “loan limit reached” message to UI

calls the borrowing completed process

**Postconditions:**

book not found – control and UI state remains SCANNING, no loan created

book not available – control and UI state remains SCANNING, no loan created

book available and patron would remain under loan limit if borrowed

* Pending loan created, added to pending loan list
* UI and control state remains SCANNING

book available and patron would reach loan limit if borrowed

* Pending loan created, added to pending loan list
* Borrowing completed process called
* UI and control state in FINALISING state

**void** borrowingCompleted();

**Preconditions:** UI and control in SCANNING state, control has reference to patron

if no pending loans have been issued

calls the cancel process, exits

for each pending loan

displays book information for each loan to the UI

Sets UI state to FINALISING

Transitions control class to FINALISING state.

**Postconditions:**

No pending loans – control and UI state CANCELLED

Pending loans exist – control and UI state FINALISING

\*Loans state is still PENDING

\*Books state is still AVAILABLE

**void** commitLoans();

**Preconditions:** UI and control in FINALISING state, control has reference to patron, pending loans exist,

All books associated with pending loans are still AVAILABLE

for each pending loan

commits the loan to the library

displays a loan slip message to the UI

for each pending loan

displays loan information to the UI

Sets UI state to COMPLETED

Transitions control class to COMPLETED state.

**Postconditions:**

Control and UI state COMPLETED

\*All Loans added to library current loan list

\*All Loans added to library full loan list

\*All Loans added to patron borrowing record

\*All Loans state is CURRENT

\*All Books state is ON\_LOAN

**void** cancel();

**Preconditions:** UI and control not in COMPLETED state

Sets UI state to CANCELLED

Transitions control class to CANCELLED state.

**Postconditions:** UI and control class in CANCELLED state.

BorrowBookUI

Constructor

Sets UI state to INITIALISED

Calls control setUI method with self as parameter.

**public** **void** run() {

Loops performing state specific input/output actions as set by control class.

While not finished

If state is READY

Prompts patron to input patronId

If no patronId entered

Initiates control class cancel process

If input invalid

Displays error message

Else

Calls control class cardSwiped method with patronId as parameter

Else if state is SCANNING

Prompts patron to input bookID

If no bookId entered

Initiates control class borrowingCompleted process

If input invalid

Displays error message

Else

Calls control class bookScanned method with pbookId as parameter

Else if state is RESTRICTED

Prompts patron to cancel

Initiates control class cancel process

Else if state is FINALISING

Prompts patron to commit loans

If patron decides not to commit loans

Initiates control class cancel process

Else

Calls control class commitLoans method

Else if state is COMPLETED

Prompts patron to complete

Displays completed message

Sets finished to TRUE

Else

Displays unhandled state message.

Throws a runtime exception

**public** **void** display(Object object) {

Displays the designated object to the user

(Assumes object has a ‘toString’ method.)

**void** setSwiping();

Throws a Runtime error if UI state is not INITIALISED  
Otherwise sets UI state to SWIPING

**void** setScanning();

Throws a Runtime error if UI state is not SWIPING  
Otherwise sets UI state to SCANNING

**void** setRestricted();

Throws a Runtime error if UI state is not SWIPING  
Otherwise sets UI state to RESTRICTED

**void** setFinalising();

Throws a Runtime error if UI state is not SCANNING  
Otherwise sets UI state to FINALISING

**void** setCompleted();

Throws a Runtime error if UI state is not FINALISING  
Otherwise sets UI state to COMPLETED

**void** setCancelled();

Throws a Runtime error if UI state is COMPLETED  
Otherwise sets UI state to CANCELLED