

OO Mobile Media Architecture (Release 3)

COMPONENT: AlbumListScreen

Required Interface: ControlPhoto

```
newPhotoAlbum(); // "New Photo Album"
deleteAlbum(); // "Delete Album"
viewAlbum(); // "Select"
```

Provided Interface: ManageAlbumList

```
append(String albumName);
delete(int i);
repaintListAlbum(String[] names);
```

COMPONENT: AddPhotoToAlbumScreen

Required Interface: ControlPhoto

```
saveNewPhoto(); // "Save Add Photo"
cancel();
```

Provided Interface: ManagePhoto

```
String getPhotoName();
String getPath();
```

COMPONENT: NewLabelScreen

Required Interface: ControlLabel

```
saveLabel(); // "Save"
cancel();
```

Required Interface: ControlPhoto

```
saveNewPhotoAlbum(); // "Save"
cancel();
```

Provided Interface: ManageLabel

```
String getLabelName();
int getFormType();
```

COMPONENT: PhotoListScreen

Required Interface: ControlPhoto

```
viewPhoto(); // "View"
newPhoto(); // "Add"
deletePhoto(); // "Delete"
editLabel(); // "Edit Label"
sortPhotos(); // "Sort by Views"
back(); // "Back"
setFavourite(); // "Set Favorite"
viewFavourite(); // "View Favorites"
```

Provided Interface: ManagePhotoList

```
append(String imageLabel);
```

COMPONENT: PhotoViewScreen

Required Interface: ControlPhoto

```
back(); // "Back"
```

COMPONENT: BaseController

Provided Interface: ControlPhoto

```
newPhotoAlbum(); // "New Photo Album"
deleteAlbum(); // "Delete Album"
saveNewPhotoAlbum(); // "Save"
viewAlbum(); // "Select"
viewPhoto(); // "View"
newPhoto(); // "Add"
saveNewPhoto(); // "Save Add Photo"
deletePhoto(); // "Delete"
editLabel(); // "Edit Label"
sortPhotos(); // "Sort by Views"
setFavourite(); // "Set Favorite"
viewFavourite(); // "View Favorites"
cancel(); // "Cancel"
back(); // "Back"
```

Provided Interface: ManageImage

```
updateImage(ImageData image);
showImageList(String recordName, boolean sort, boolean favorite);
```

Provided Interface: ManageScreen

```
setCurrentScreenName(String currentScreenName);
```

Required Interface: SetScreen

```
setScreen(NewLabelScreen screen);
```

Required Interface: ManageAlbumList

```
append(String albumName);
delete();
repaintListAlbum(String[] names);
```

Required Interface: ManagePhoto

```
String getPhotoName();
String getPath();
```

Required Interface: ManageLabel

```
String getLabelName();
int getFormType();
```

Required Interface: ManagePhotoList

```
append(String imageLabel);
```

Required Interface: ManagePhotoInfo

```
addNewPhotoToAlbum(String label, String path, String album) throws InvalidImageDataException, PersistenceMechanismException;
createNewPhotoAlbum(String albumName) throws PersistenceMechanismException, InvalidPhotoAlbumNameException;
deleteImage(String imageName, String storeName) throws PersistenceMechanismException, ImageNotFoundException;
deletePhotoAlbum(String albumName) throws PersistenceMechanismException;
String[] getAlbumNames();
Image getImageFromRecordStore(String recordStore, String imageName) throws ImageNotFoundException, PersistenceMechanismException;
ImageData[] getImages(String recordName) throws UnavailablePhotoAlbumException;
resetImageData() throws PersistenceMechanismException;
ImageData getImageInfo(String imageName) throws ImageNotFoundException, NullAlbumDataReference;
boolean updateImageInfo(ImageData oldData, ImageData newData) throws InvalidImageDataException, PersistenceMechanismException;
```

COMPONENT: PhotoController

Provided Interface: ControlLabel

```
saveLabel(); // "Save"
cancel();
```

Provided Interface: SetScreen

```
setScreen(NewLabelScreen screen);
```

Required Interface: ManageLabel

```
String getLabelName();
```

Required Interface: ManageImage

```
updateImage(ImageData image);
showImageList(String recordName, boolean sort, boolean favorite);
```

Required Interface: ManageScreen

```
setCurrentScreenName(String currentScreenName);
```

COMPONENT: AlbumData

Provided Interface: ManagePhotoInfo

```
addNewPhotoToAlbum(String label, String path, String album) throws InvalidImageDataException, PersistenceMechanismException;
createNewPhotoAlbum(String albumName) throws PersistenceMechanismException, InvalidPhotoAlbumNameException;
deleteImage(String imageName, String storeName) throws PersistenceMechanismException, ImageNotFoundException;
deletePhotoAlbum(String albumName) throws PersistenceMechanismException;
String[] getAlbumNames();
Image getImageFromRecordStore(String recordStore, String imageName) throws ImageNotFoundException, PersistenceMechanismException;
Hashtable getImageInfoTable();
ImageData[] getImages(String recordName) throws UnavailablePhotoAlbumException;
resetImageData() throws PersistenceMechanismException;
ImageData getImageInfo(String imageName) throws ImageNotFoundException, NullAlbumDataReference;
boolean updateImageInfo(ImageData oldData, ImageData newData) throws InvalidImageDataException, PersistenceMechanismException;
```

Required Interface: PersistPhoto

```
addImageData(String photoname, String path, String albumname) throws InvalidImageDataException, PersistenceMechanismException;
createNewPhotoAlbum(String albumName) throws PersistenceMechanismException, InvalidPhotoAlbumNameException;
deletePhotoAlbum(String albumName) throws PersistenceMechanismException;
boolean deleteSingleImageFromRMS(String storeName, String imageName) throws PersistenceMechanismException, ImageNotFoundException, NullAlbumDataReference;
String[] getAlbumNames();
ImageData getImageInfo(String imageName) throws ImageNotFoundException, NullAlbumDataReference;
```

```
loadAlbums() throws InvalidImageDataException, PersistenceMechanismException;
ImageData[] loadImageDataFromRMS(String recordName) throws PersistenceMechanismException, InvalidImageDataException;
Image loadSingleImageFromRMS(String recordName, String imageName, int recordId) throws PersistenceMechanismException;
resetImageRecordStore() throws InvalidImageDataException, PersistenceMechanismException;
boolean updateImageInfo(ImageData oldData, ImageData newData) throws InvalidImageDataException, PersistenceMechanismException;
```

COMPONENT: ImageAcessor

Provided Interface: PersistPhoto

```
addImageData(String photoname, String path, String albumname) throws InvalidImageDataException, PersistenceMechanismException;
createNewPhotoAlbum(String albumName) throws PersistenceMechanismException, InvalidPhotoAlbumNameException;
deletePhotoAlbum(String albumName) throws PersistenceMechanismException;
boolean deleteSingleImageFromRMS(String storeName, String imageName) throws PersistenceMechanismException, ImageNotFoundException, NullAlbumDataReference;
String[] getAlbumNames();
ImageData getImageInfo(String imageName) throws ImageNotFoundException, NullAlbumDataReference;
loadAlbums() throws InvalidImageDataException, PersistenceMechanismException;
ImageData[] loadImageDataFromRMS(String recordName) throws PersistenceMechanismException, InvalidImageDataException;
Image loadSingleImageFromRMS(String recordName, String imageName, int recordId) throws PersistenceMechanismException;
resetImageRecordStore() throws InvalidImageDataException, PersistenceMechanismException;
boolean updateImageInfo(ImageData oldData, ImageData newData) throws InvalidImageDataException, PersistenceMechanismException;
```

Required Interface: ManagePhotoInfo

```
Hashtable getImageInfoTable();
```