

Resource Code

This appendix defines the routines used by the Venn Diagrammer application to create, read, and write the resources it uses to store the user's preferences. The application expects to find those resources in a file named "Venn Diagrammer Preferences" in the Preferences folder in the currently-active System folder. If no such file is found, the application creates a new file of the desired name in that location; then it copies into that file a default set of preferences settings that is contained in the application's resource file.

```

UNIT Preferences;
INTERFACE
    USES
        Folders, Global, Utilities;

    CONST
        kPrefResType =    'PRFN';    {type of preferences resource}
        kPrefResID =      259;        {resource ID of preferences resource}

    TYPE
        {structure of a resource that contains Venn diagram preferences}
        MyPrefsRec = RECORD
            autoDiag:    Boolean;    {do we automatically fix the diagram?}
            showName:    Boolean;    {do we show names of valid arguments?}
            isImport:    Boolean;    {do subjects have existential import?}
            isRandom:    Boolean;    {do we select next setting randomly?}
            emptyInd:    Integer;    {index of the desired emptiness pattern}
            existInd:    Integer;    {index of the desired existence symbol}
        END;
        MyPrefsPtr = ^MyPrefsRec;
        MyPrefsHnd = ^MyPrefsPtr;

    FUNCTION DoCopyResource (rType: ResType; rID: Integer; source: Integer;
                                dest: Integer): OSErr;

    PROCEDURE DoReadPrefs;
    PROCEDURE DoSavePrefs;

IMPLEMENTATION

{DoCopyResource}
{copy a resource from one open resource file [source] to another [dest];}
{make sure not to alter the current resource file }

```

Resource Code

```

{ and to preserve resource attributes}
FUNCTION DoCopyResource (rType: ResType; rID: Integer; source: Integer;
                        dest: Integer): OSErr;

VAR
    myHandle:   Handle;           {handle to resource to copy}
    myName:     Str255;           {name of resource to copy}
    myAttr:     Integer;          {resource attributes}
    myType:     ResType;          {ignored; used for GetResInfo}
    myID:       Integer;          {ignored; used for GetResInfo}
    myResult:   OSErr;
    myCurrent:  Integer;          {current resource file on entry}
BEGIN
    myCurrent := CurResFile;      {remember current resource file}
    UseResFile(source);           {set the source resource file}
    myHandle := Get1Resource(rType, rID); {open the source resource}
    IF myHandle <> NIL THEN
        BEGIN
            GetResInfo(myHandle, myID, myType, myName); {get res name}
            myAttr := GetResAttrs(myHandle);             {get res attributes}
            DetachResource(myHandle);                    {so we can copy the resource}
            UseResFile(dest);                            {set destination resource file}
            IF ResError = noErr THEN
                AddResource(myHandle, rType, rID, myName);
            IF ResError = noErr THEN
                SetResAttrs(myHandle, myAttr); {set res attributes of copy}
            IF ResError = noErr THEN
                ChangedResource(myHandle);    {mark resource as changed}
            IF ResError = noErr THEN
                WriteResource(myHandle);      {write resource data}
        END;

        DoCopyResource := ResError;          {return result code}
        ReleaseResource(myHandle);           {get rid of resource data}
        UseResFile(myCurrent);               {restore original resource file}
    END;

{DoCreatePrefsFile:}
{Create a preferences file in the specified location.}
{The initial settings are just those in the app's resource file.}
FUNCTION DoCreatePrefsFile (myVRefNum: Integer; myDirID: LongInt;
                            myName: Str255): Integer;

VAR
    myResNum:   Integer;

```

Resource Code

```

myResult:    OSErr;
myID:        Integer;    {resource ID of resource in app's res fork}
myHandle:    Handle;      {handle to resource in app's res fork}
myType:      ResType;     {ignored; used for GetResInfo}
BEGIN
    myResult := noErr;
    HCreateResFile(myVRefNum, myDirID, myName);
    IF ResError = noErr THEN
        BEGIN
            myResNum := HOpenResFile(myVRefNum, myDirID, myName, fsCurPerm);
            IF myResNum <> -1 THEN
                BEGIN
                    UseResFile(gAppsResourceFile);
                    myHandle := Get1Resource(kPrefResType, kPrefResID);
                    IF ResError = noErr THEN
                        BEGIN
                            GetResInfo(myHandle, myID, myType, myName);
                            myResult := DoCopyResource(kPrefResType, myID,
                                                         gAppsResourceFile, myResNum);
                        END
                    ELSE
                        BEGIN
                            CloseResFile(myResNum);
                            myResult := HDelete(myVRefNum, myDirID, myName);
                            myResNum := -1;
                        END;
                    END;
                END;
            END;
        END;
        DoCreatePrefsFile := myResNum;
    END;
END; {DoCreatePrefsFile}

{DoReadPrefs:}
{Open the application's global preferences file and read indicated settings.}
PROCEDURE DoReadPrefs;
VAR
    myVRefNum: Integer;
    myDirID:   LongInt;
    myName:    Str255;    {name of this application}
    myPrefs:   Handle;    {handle to actual preferences data}
    myResNum:  Integer;    {reference number of opened resource file}
    myResult:  OSErr;
CONST

```

Resource Code

```

        kNameID = 4000;           {resource ID of 'STR#' with filename}
BEGIN
    {Determine the name of the preferences file.}
    GetIndString(myName, kNameID, 1);

    {Figure out where the preferences file is.}
    IF IsFindFolder THEN
        myResult := FindFolder(kOnSystemDisk, kPreferencesFolderType,
                                kDontCreateFolder, myVRefNum, myDirID)
    ELSE
        myResult := -1;

    IF myResult <> noErr THEN
        BEGIN
            myVRefNum := 0;           {use default volume}
            myDirID := 0;           {use default directory}
        END;

    {Open the preferences resource file.}
    myResNum := HOpenResFile(myVRefNum, myDirID, myName, fsCurPerm);

    {If no preferences file successfully opened, create one }
    { by copying default preferences in app's resource file.}
    IF myResNum = -1 THEN
        myResNum := DoCreatePrefsFile(myVRefNum, myDirID, myName);

    IF myResNum <> -1 THEN           {if we successfully opened the file...}
        BEGIN
            UseResFile(myResNum);    {make the new resource file current one}
            myPrefs := Get1Resource(kPrefResType, kPrefResID);
            IF myPrefs = NIL THEN
                exit(DoReadPrefs);
            WITH MyPrefsHnd(myPrefs)^ DO
                BEGIN
                    {read the preferences settings}
                    gAutoAdjust := autoDiag;
                    gShowNames := showName;
                    gGiveImport := isImport;
                    gStepRandom := isRandom;
                    gEmptyIndex := emptyInd;
                    gExistIndex := existInd;
                END;

            {Make sure some preferences globals make sense.}

```

Resource Code

```

    IF NOT (gExistIndex IN [1..4]) THEN
        gExistIndex := 1;
    IF NOT (gEmptyIndex IN [1..4]) THEN
        gEmptyIndex := 1;

    {Reinstate the application's resource file.}
    UseResFile(gAppsResourceFile);
END;

gPreferencesFile := myResNum;           {remember its resource ID}
END; {DoReadPrefs}

{DoSavePrefs:}
{Save the current preference settings.}
PROCEDURE DoSavePrefs;
    VAR
        myPrefData: Handle;      {handle to new resource data}
        myHandle:   Handle;      {handle to resource to replace}
        myName:     Str255;      {name of resource to copy}
        myAttr:     Integer;      {resource attributes}
        myType:     ResType;      {ignored; used for GetResInfo}
        myID:       Integer;      {ignored; used for GetResInfo}
BEGIN
    {Make sure we have an open preferences file.}
    IF gPreferencesFile = -1 THEN
        exit(DoSavePrefs);

    myPrefData := NewHandleClear(sizeof(MyPrefsRec));
    HLock(myPrefData);
    WITH MyPrefsHnd(myPrefData)^ DO
        BEGIN
            autoDiag := gAutoAdjust;
            showName := gShowNames;
            isImport := gGiveImport;
            isRandom := gStepRandom;
            emptyInd := gEmptyIndex;
            existInd := gExistIndex;
        END;

    UseResFile(gPreferencesFile);           {use preferences file}
    myHandle := Get1Resource(kPrefResType, kPrefResID);
    IF myHandle <> NIL THEN
        BEGIN

```

Resource Code

```

    GetResInfo(myHandle, myID, myType, myName);    {get res name}
    myAttr := GetResAttrs(myHandle);               {get res attributes}
    RmveResource(myHandle);
    IF ResError = noErr THEN
        AddResource(myPrefData, kPrefResType, kPrefResID, myName);
    IF ResError = noErr THEN
        WriteResource(myPrefData);
END;

HUnlock(myPrefData);
ReleaseResource(myPrefData);
UseResFile(gAppsResourceFile);                   {restore app's resource file}

END; {DoSavePrefs}

END. {UNIT Preferences}

```