

Debugging in Xcode

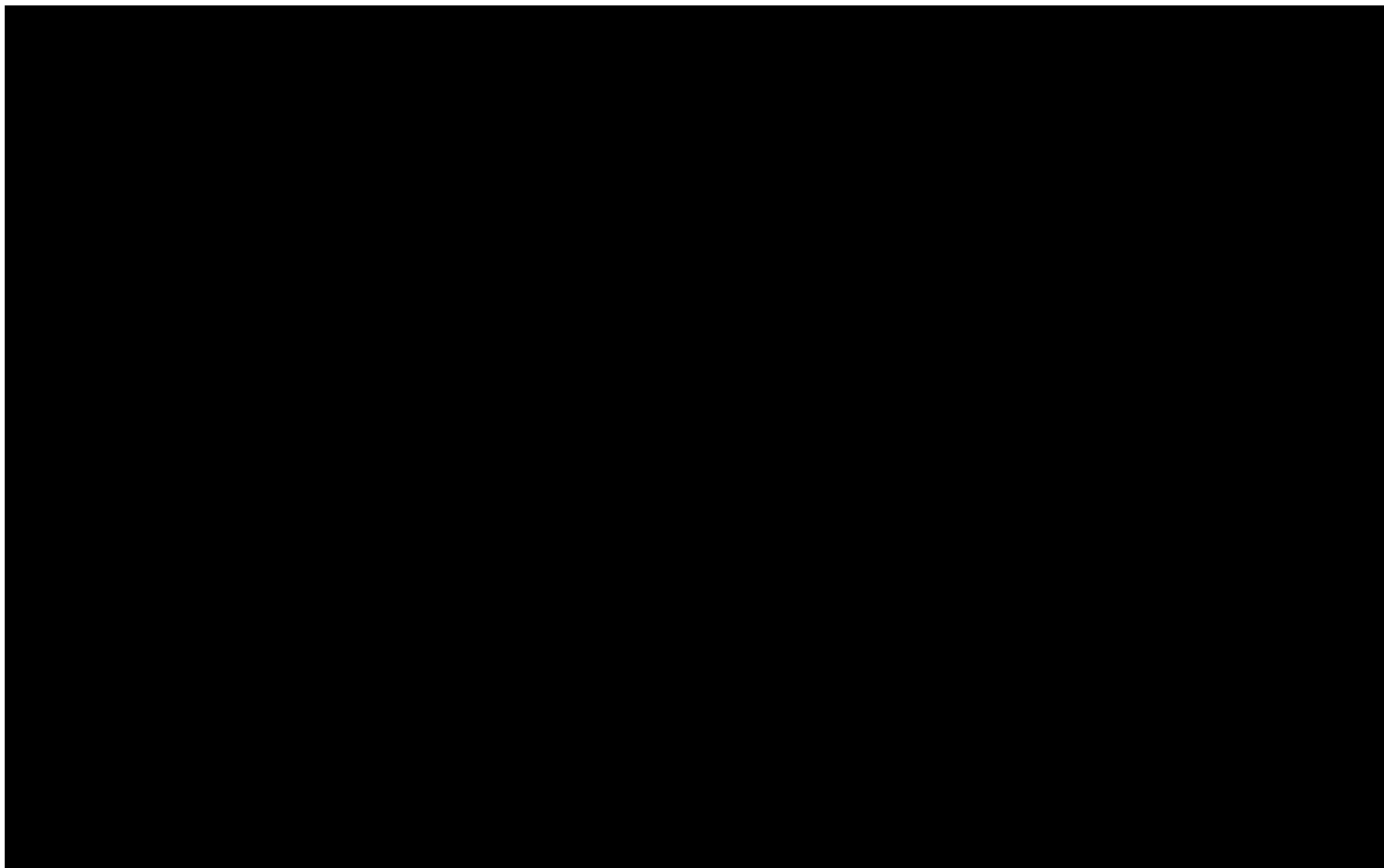
Session 412

Ken Orr

Xcode Debugger UI Manager

These are confidential sessions—please refrain from streaming, blogging, or taking pictures







Control Flow



Correctness

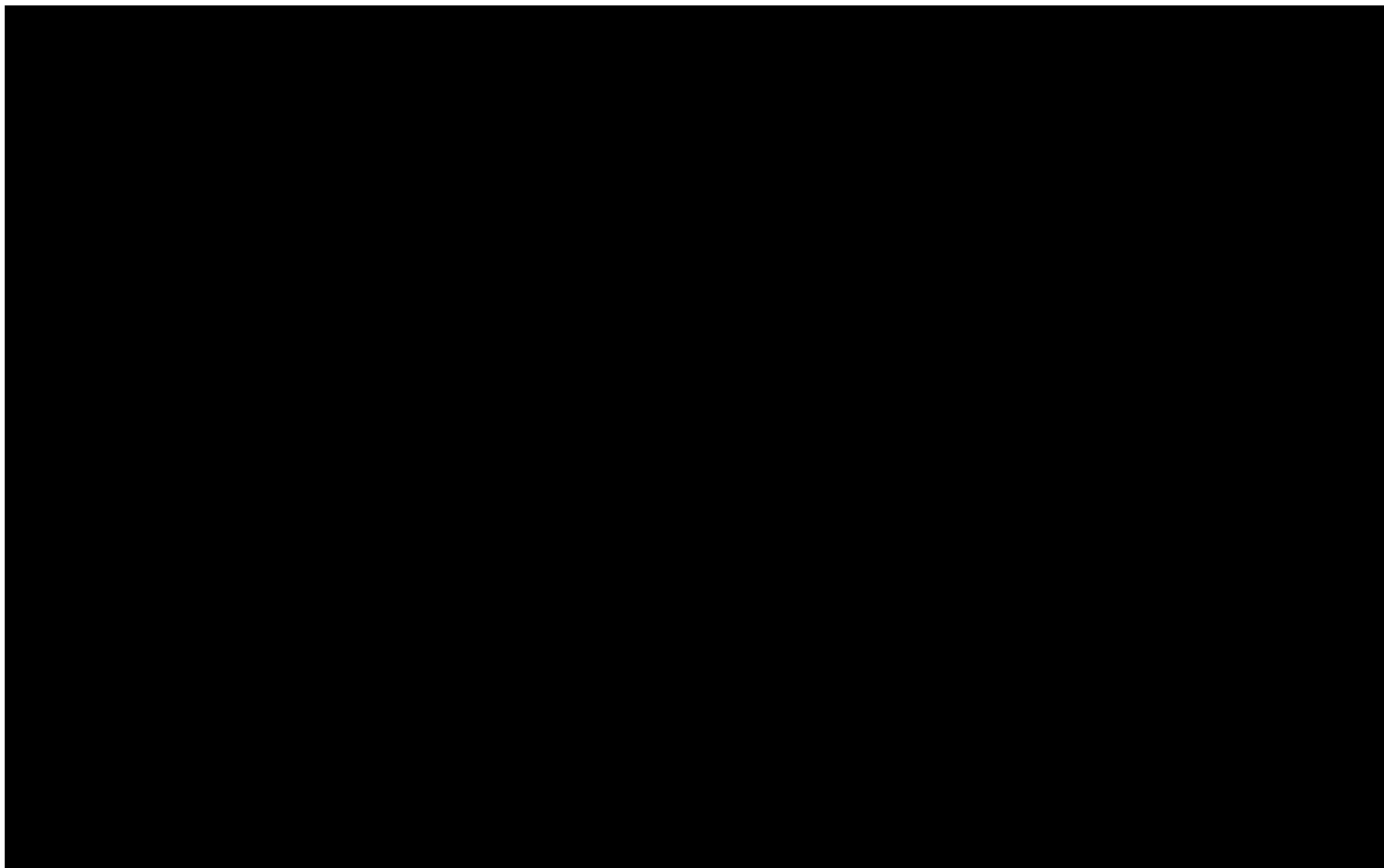


Breakpoints

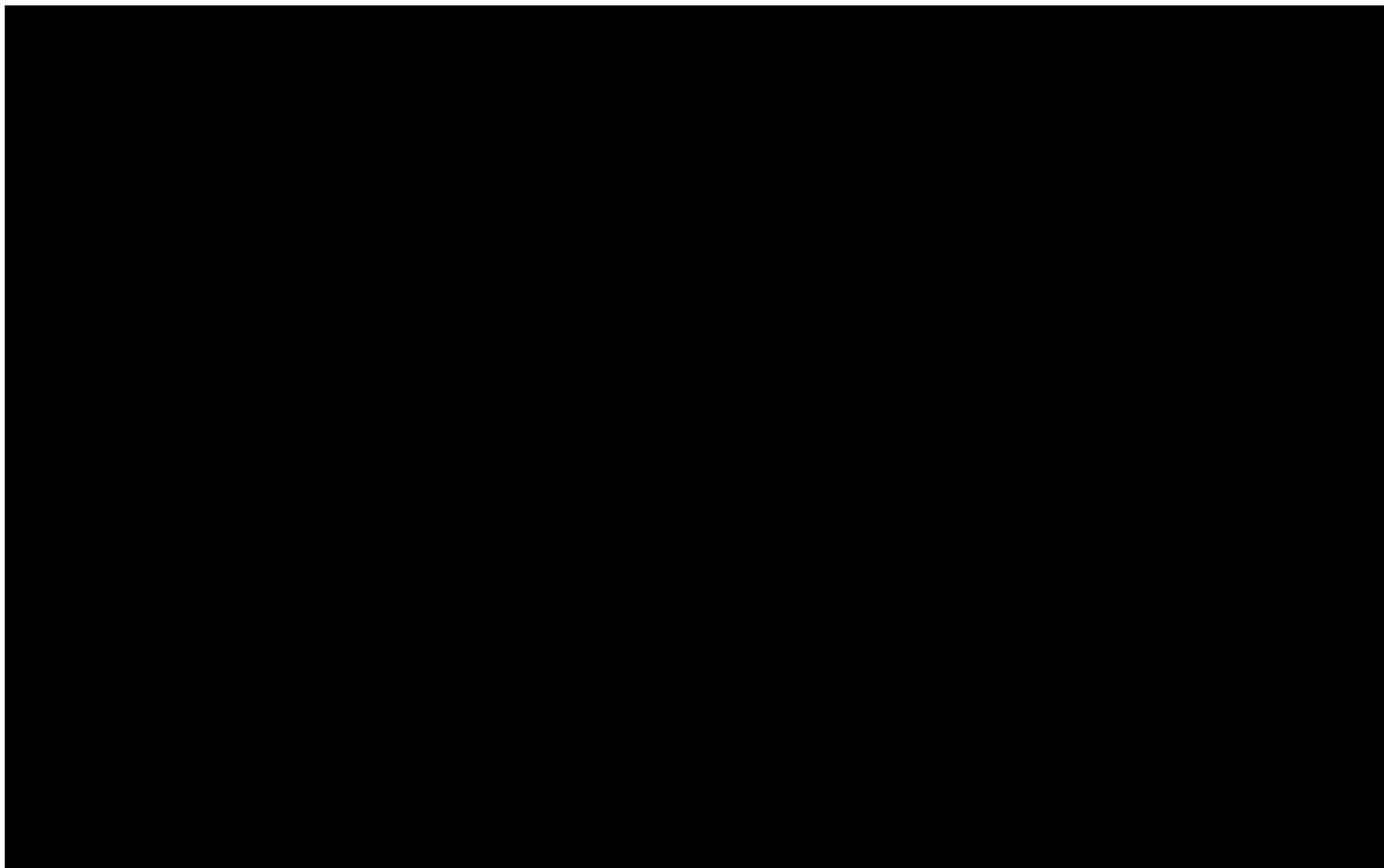


Variables

Breakpoints and Breakpoint Actions



```
NSLog(@"%ld", foo);
```

Creating and Deleting Breakpoints

The screenshot shows the Xcode IDE interface. At the top, the workspace is named "Sketch.xcworkspace" and the current file is "SKTWindowController.m". A status bar indicates "Build Sketch: Succeeded" on 5/25/12 at 8:50 AM. The toolbar shows "Run", "Stop", "Scheme", and "Breakpoints" buttons. The left sidebar displays the project structure for "Sketch", with "SKTWindowController.m" selected under the "Controllers" folder. The main editor window shows the Objective-C code for the SKTWindowController class, with a breakpoint (M) set on the dealloc method. The code includes comments and method implementations for initWithWindowNibName:, dealloc, and dealloc.

```
21
22
23
24
25 - (id)init {
26
27     // Do the regular Cocoa thing, specifying a particular nib.
28     self = [super initWithWindowNibName:@"DrawWindow"];
29     if (self) {
30
31         // Create a grid for use by graphic views whose "grid" property is bound to this object
32         _grid = [[SKTGrid alloc] init];
33
34         // Set the zoom factor to a reasonable default (100%).
35         _zoomFactor = 1.0f;
36
37     }
38     return self;
39 }
40
41
42
43 - (void)dealloc {
44
45     // Stop observing the tool palette.
46     [[NSNotificationCenter defaultCenter] removeObserver:self name:SKTSelectedToolDidChange
47         [SKTToolPaletteController sharedToolPaletteController]];
48     // Stop observing the document's canvas size.
```

Creating and Deleting Breakpoints

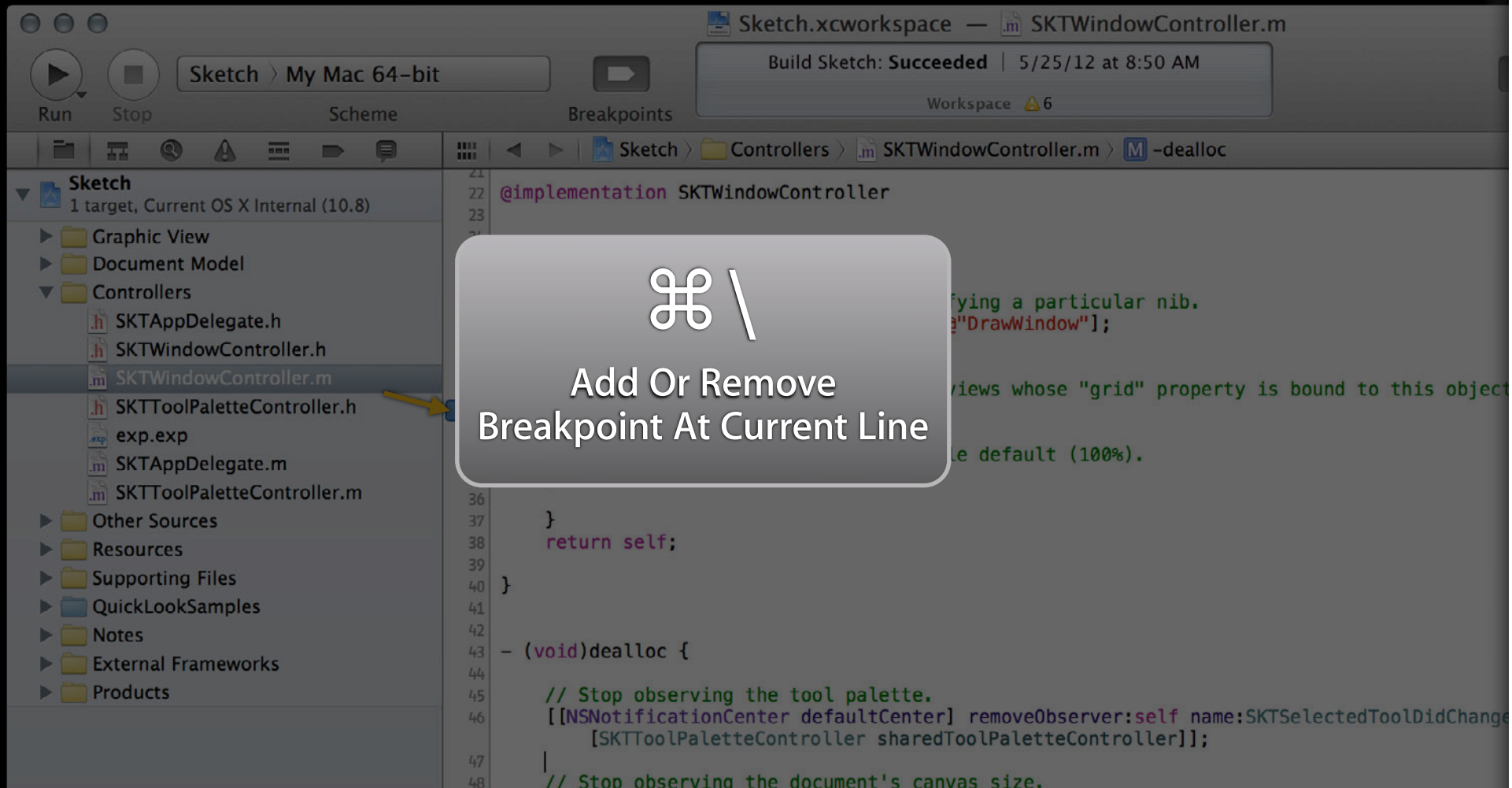
The screenshot shows the Xcode IDE interface. At the top, the title bar reads "Sketch.xcworkspace — SKTWindowController.m". Below the title bar, there are buttons for "Run", "Stop", "Scheme", and "Breakpoints". A status bar indicates "Build Sketch: Succeeded | 5/25/12 at 8:50 AM" and "Workspace ⚠️ 6".

The left sidebar shows a project tree for "Sketch" with a target "Current OS X Internal (10.8)". The "Controllers" folder is expanded, showing files: SKTAppDelegate.h, SKTWindowController.h, SKTWindowController.m (selected), SKTToolPaletteController.h, exp.exp, SKTAppDelegate.m, and SKTToolPaletteController.m. A yellow arrow points to the selected file.

The main editor window displays the source code for SKTWindowController.m. A blue breakpoint icon is set on line 32, which is highlighted. The code is as follows:

```
21
22 @implementation SKTWindowController
23
24
25 - (id)init {
26
27     // Do the regular Cocoa thing, specifying a particular nib.
28     self = [super initWithWindowNibName:@"DrawWindow"];
29     if (self) {
30
31         // Create a grid for use by graphic views whose "grid" property is bound to this object
32         _grid = [[SKTGrid alloc] init];
33
34         // Set the zoom factor to a reasonable default (100%).
35         _zoomFactor = 1.0f;
36
37     }
38     return self;
39 }
40
41
42
43 - (void)dealloc {
44
45     // Stop observing the tool palette.
46     [[NSNotificationCenter defaultCenter] removeObserver:self name:SKTSelectedToolDidChange
47         [SKTToolPaletteController sharedToolPaletteController]];
48     // Stop observing the document's canvas size.
```

Creating and Deleting Breakpoints



The screenshot shows the Xcode IDE interface. The top toolbar includes buttons for Run, Stop, Scheme, and Breakpoints. A status bar at the top right indicates "Build Sketch: Succeeded" on 5/25/12 at 8:50 AM. The left sidebar displays a project tree for "Sketch", with "SKTWindowController.m" selected under the "Controllers" folder. The main editor window shows the source code for "SKTWindowController.m", with the current line being line 46: `[[NSNotificationCenter defaultCenter] removeObserver:self name:SKTSelectedToolDidChange`. A callout box with a white background and rounded corners is overlaid on the code, containing a breakpoint icon (a circle with a cross) and the text "Add Or Remove Breakpoint At Current Line". An arrow points from the callout box to the breakpoint icon in the left margin of the code editor.

Sketch > My Mac 64-bit

Build Sketch: Succeeded | 5/25/12 at 8:50 AM

Workspace ⚠ 6

Sketch > Controllers > SKTWindowController.m > M -dealloc

Sketch

- 1 target, Current OS X Internal (10.8)
- Graphic View
- Document Model
- Controllers
 - SKTAppDelegate.h
 - SKTWindowController.h
 - SKTWindowController.m
 - SKTToolPaletteController.h
 - exp.exp
 - SKTAppDelegate.m
 - SKTToolPaletteController.m
- Other Sources
- Resources
- Supporting Files
- QuickLookSamples
- Notes
- External Frameworks
- Products

21 @implementation SKTWindowController

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37 }

38 return self;

39

40 }

41

42

43 - (void)dealloc {

44

45 // Stop observing the tool palette.

46 [[NSNotificationCenter defaultCenter] removeObserver:self name:SKTSelectedToolDidChange

47 |

48 // Stop observing the document's canvas size.

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

1000

1001

1002

1003

1004

1005

1006

1007

1008

1009

1010

1011

1012

1013

1014

1015

1016

1017

1018

1019

1020

1021

1022

1023

1024

1025

1026

1027

1028

1029

1030

1031

1032

1033

1034

1035

1036

1037

1038

1039

1040

1041

1042

1043

1044

1045

1046

1047

1048

1049

1050

1051

1052

1053

1054

1055

1056

1057

1058

1059

1060

1061

1062

1063

1064

1065

1066

1067

1068

1069

1070

1071

1072

1073

1074

1075

1076

1077

1078

1079

1080

1081

1082

1083

1084

1085

1086

1087

1088

1089

1090

1091

1092

1093

1094

1095

1096

1097

1098

1099

1100

1101

1102

1103

1104

1105

1106

1107

1108

1109

1110

1111

1112

1113

1114

1115

1116

1117

1118

1119

1120

1121

1122

1123

1124

1125

1126

1127

1128

1129

1130

1131

1132

1133

1134

1135

1136

1137

1138

1139

1140

1141

1142

1143

1144

1145

1146

1147

1148

1149

1150

1151

1152

1153

1154

1155

1156

1157

1158

1159

1160

1161

1162

1163

1164

1165

1166

1167

1168

1169

1170

1171

1172

1173

1174

1175

1176

1177

1178

1179

1180

1181

1182

1183

1184

1185

1186

1187

1188

1189

1190

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

1201

1202

1203

1204

1205

1206

1207

1208

1209

1210

1211

1212

1213

1214

1215

1216

1217

1218

1219

1220

1221

1222

1223

1224

1225

1226

1227

1228

1229

1230

1231

1232

1233

1234

1235

1236

1237

1238

1239

1240

1241

1242

1243

1244

1245

1246

1247

1248

1249

1250

1251

1252

1253

1254

1255

1256

1257

1258

1259

1260

1261

1262

1263

1264

1265

1266

1267

1268

1269

1270

1271

1272

1273

1274

1275

1276

1277

1278

1279

1280

1281

1282

1283

1284

1285

1286

1287

1288

1289

1290

1291

1292

1293

1294

1295

1296

1297

1298

1299

1300

1301

1302

1303

1304

1305

1306

1307

1308

1309

1310

1311

1312

1313

1314

1315

1316

1317

1318

1319

1320

1321

1322

1323

1324

1325

1326

1327

1328

1329

1330

1331

1332

1333

1334

1335

1336

1337

1338

1339

1340

1341

1342

1343

1344

1345

1346

1347

1348

1349

1350

1351

1352

1353

1354

1355

1356

1357

1358

1359

1360

1361

1362

1363

1364

1365

1366

1367

1368

1369

1370

1371

1372

1373

1374

1375

1376

1377

1378

1379

1380

1381

1382

1383

1384

1385

1386

1387

1388

1389

1390

1391

1392

1393

1394

1395

1396

1397

1398

1399

1400

1401

1402

1403

1404

1405

1406

1407

1408

1409

1410

1411

1412

1413

1414

1415

1416

1417

1418

1419

1420

1421

1422

1423

1424

1425

1426

1427

1428

1429

1430

1431

1432

1433

1434

1435

1436

1437

1438

1439

1440

1441

1442

1443

1444

1445

1446

1447

14

Creating and Deleting Breakpoints

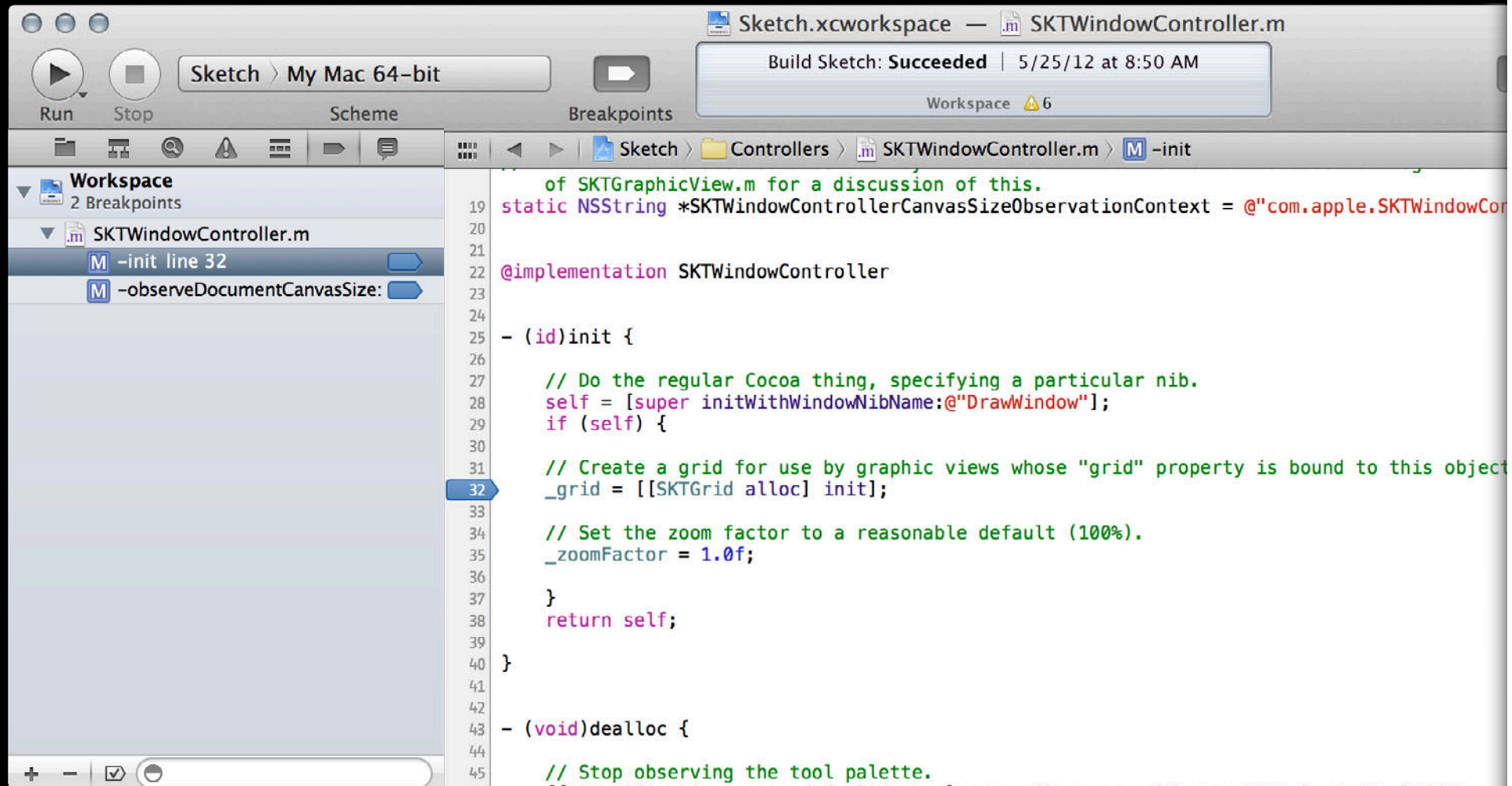
The screenshot shows the Xcode IDE interface. At the top, the title bar reads "Sketch.xcworkspace — SKTWindowController.m". Below the title bar, there are controls for "Run", "Stop", "Scheme" (set to "Sketch > My Mac 64-bit"), and "Breakpoints". A status bar indicates "Build Sketch: Succeeded | 5/25/12 at 8:50 AM" and "Workspace ⚠️ 6".

The left sidebar shows a project tree for "Sketch" with 1 target, "Current OS X Internal (10.8)". The tree includes folders for "Graphic View", "Document Model", "Controllers", "Other Sources", "Resources", "Supporting Files", "QuickLookSamples", "Notes", "External Frameworks", and "Products". Under "Controllers", the file "SKTWindowController.m" is selected.

The main editor window displays the source code for "SKTWindowController.m" with a breakpoint set on the "dealloc" method. The code is as follows:

```
21
22
23
24
25 - (id)init {
26
27     // Do the regular Cocoa thing, specifying a particular nib.
28     self = [super initWithWindowNibName:@"DrawWindow"];
29     if (self) {
30
31         // Create a grid for use by graphic views whose "grid" property is bound to this object
32         _grid = [[SKTGrid alloc] init];
33
34         // Set the zoom factor to a reasonable default (100%).
35         _zoomFactor = 1.0f;
36
37     }
38     return self;
39 }
40
41
42
43 - (void)dealloc {
44
45     // Stop observing the tool palette.
46     [[NSNotificationCenter defaultCenter] removeObserver:self name:SKTSelectedToolDidChange
47         [SKTToolPaletteController sharedToolPaletteController]];
48     // Stop observing the document's canvas size.
```


Managing Your Breakpoints

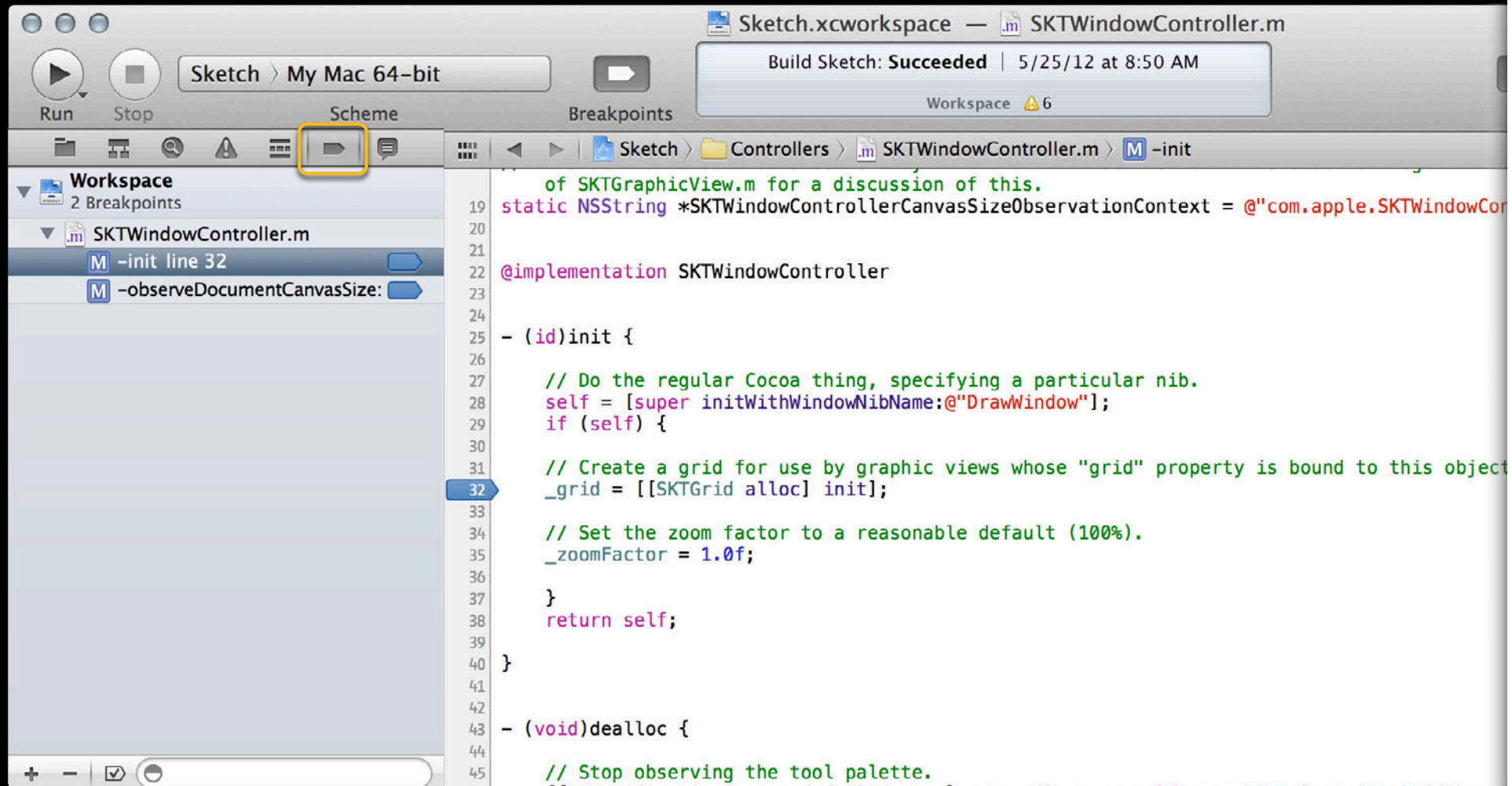


The screenshot displays the Xcode IDE interface. At the top, the title bar shows 'Sketch.xcworkspace' and 'SKTWindowController.m'. Below the title bar, there are controls for 'Run' (a play button), 'Stop' (a square button), 'Scheme' (set to 'Sketch > My Mac 64-bit'), and 'Breakpoints' (a right-pointing arrow). A status bar indicates 'Build Sketch: Succeeded | 5/25/12 at 8:50 AM' and 'Workspace ⚠️ 6'.

The main workspace is divided into two panes. The left pane, titled 'Workspace', shows a tree view with '2 Breakpoints' under the 'SKTWindowController.m' file. Two breakpoints are listed: '-init line 32' and '-observeDocumentCanvasSize:'. The right pane shows the source code for 'SKTWindowController.m' with line numbers 19 through 45. A blue highlight is on line 32, which is the line where a breakpoint is set. The code includes comments and method implementations for initialization and deallocation.

```
19 // See SKTGraphicView.m for a discussion of this.
20 static NSString *SKTWindowControllerCanvasSizeObservationContext = @"com.apple.SKWindowCon
21
22 @implementation SKTWindowController
23
24
25 - (id)init {
26
27     // Do the regular Cocoa thing, specifying a particular nib.
28     self = [super initWithWindowNibName:@"DrawWindow"];
29     if (self) {
30
31         // Create a grid for use by graphic views whose "grid" property is bound to this object
32         _grid = [[SKTGrid alloc] init];
33
34         // Set the zoom factor to a reasonable default (100%).
35         _zoomFactor = 1.0f;
36
37     }
38     return self;
39 }
40
41
42
43 - (void)dealloc {
44
45     // Stop observing the tool palette.
```

Managing Your Breakpoints



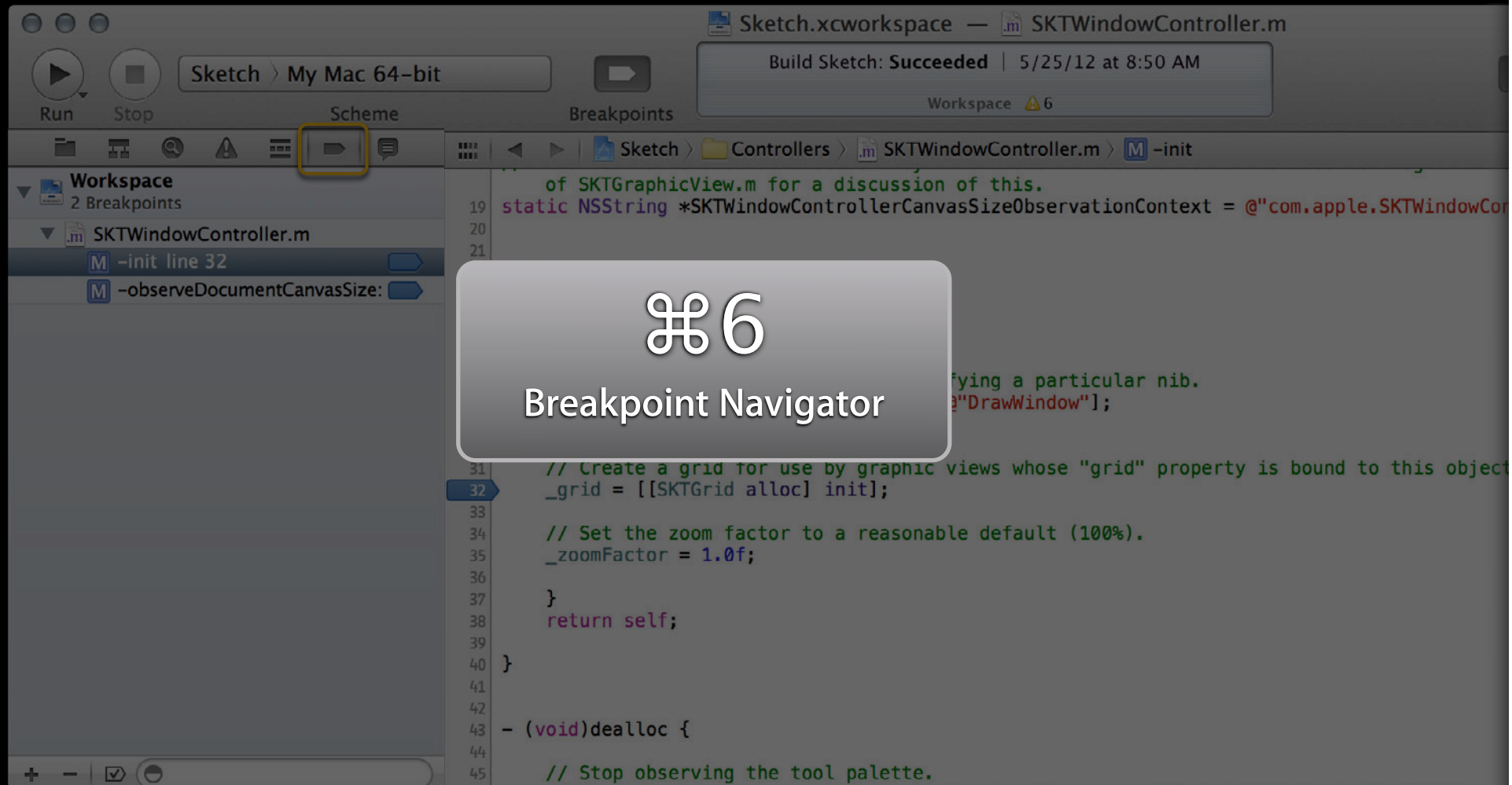
The screenshot shows the Xcode IDE interface. At the top, the title bar reads "Sketch.xcworkspace — SKTWindowController.m". Below the title bar, there are buttons for "Run", "Stop", "Scheme", and "Breakpoints". A notification box indicates "Build Sketch: Succeeded | 5/25/12 at 8:50 AM" and "Workspace ⚠️ 6".

The left sidebar shows the "Workspace" with "2 Breakpoints". Underneath, the file "SKTWindowController.m" is listed with two breakpoints: "-init line 32" and "-observeDocumentCanvasSize:". The "Scheme" button in the top toolbar is highlighted with a yellow box.

The main editor window displays the source code for "SKTWindowController.m". The code is as follows:

```
19 of SKTGraphicView.m for a discussion of this.
20 static NSString *SKTWindowControllerCanvasSizeObservationContext = @"com.apple.SKWindowCon
21
22 @implementation SKTWindowController
23
24
25 - (id)init {
26
27     // Do the regular Cocoa thing, specifying a particular nib.
28     self = [super initWithWindowNibName:@"DrawWindow"];
29     if (self) {
30
31         // Create a grid for use by graphic views whose "grid" property is bound to this object
32         _grid = [[SKTGrid alloc] init];
33
34         // Set the zoom factor to a reasonable default (100%).
35         _zoomFactor = 1.0f;
36
37     }
38     return self;
39 }
40
41
42
43 - (void)dealloc {
44
45     // Stop observing the tool palette.
```

Managing Your Breakpoints



The screenshot shows the Xcode interface with the Breakpoint Navigator on the left and a code editor on the right. The Breakpoint Navigator displays two breakpoints for the file SKTWindowController.m: one at line 32 for the method -init and another for the method -observeDocumentCanvasSize:.

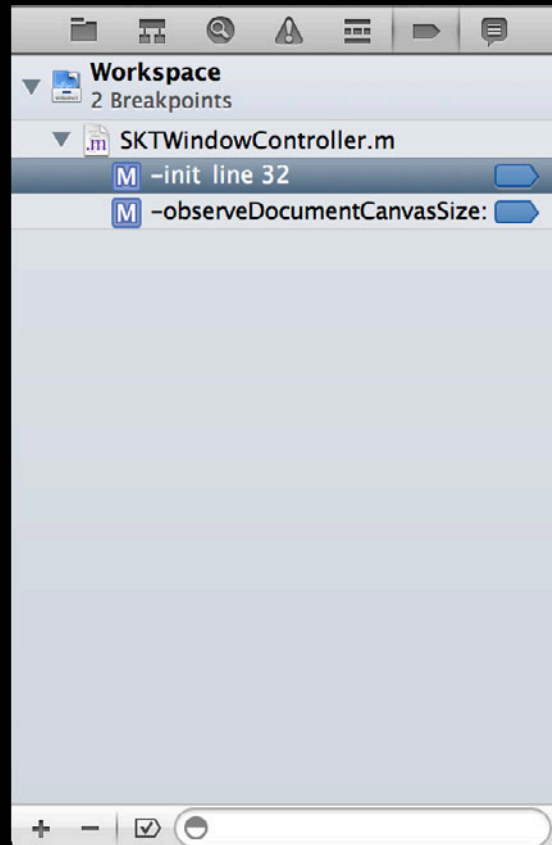
The code editor shows the following code:

```
19 static NSString *SKTWindowControllerCanvasSizeObservationContext = @"com.apple.SKWindowCon
20
21
22
23 of SKTGraphicView.m for a discussion of this.
24
25
26
27
28
29
30
31 // Create a grid for use by graphic views whose "grid" property is bound to this object
32 _grid = [[SKTGrid alloc] init];
33
34 // Set the zoom factor to a reasonable default (100%).
35 _zoomFactor = 1.0f;
36
37 }
38 return self;
39
40 }
41
42
43 - (void)dealloc {
44
45 // Stop observing the tool palette.
```

⌘6

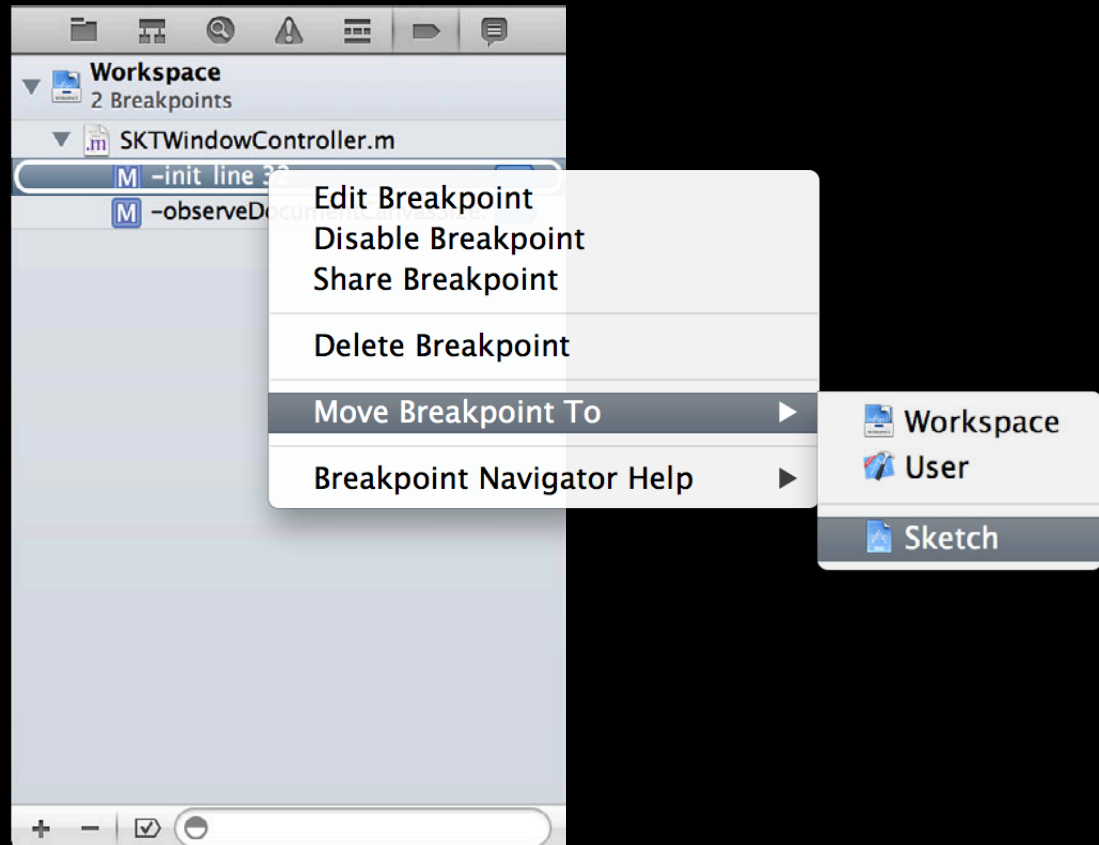
Breakpoint Navigator

Managing Your Breakpoints

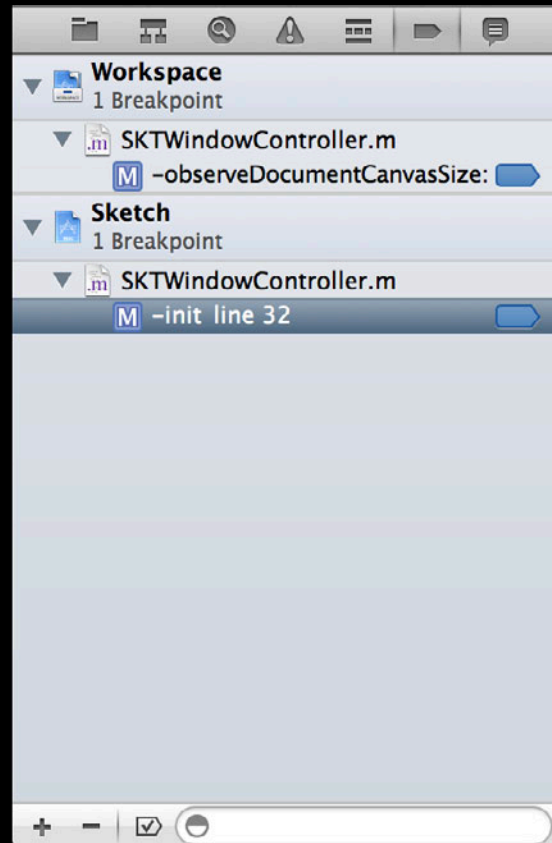


Managing Your Breakpoints

Change a Breakpoint Group
Control-click a breakpoint to
move it to a different group

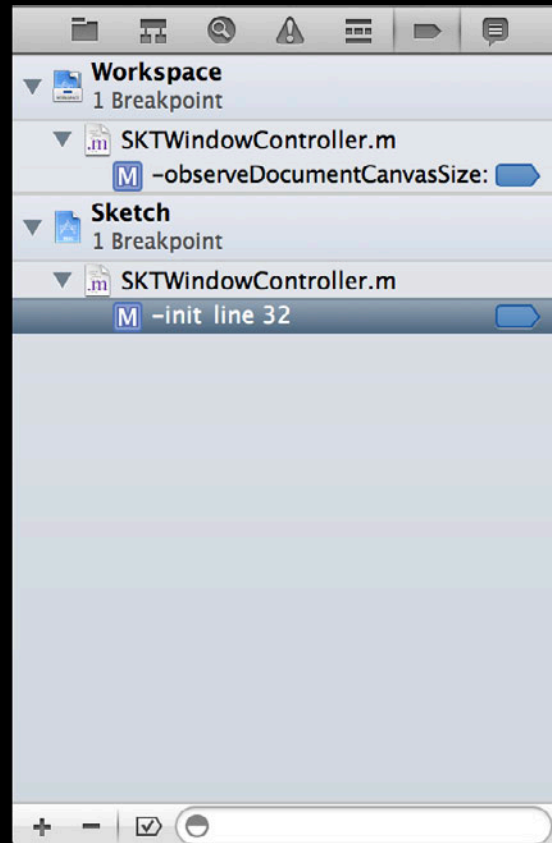


Managing Your Breakpoints



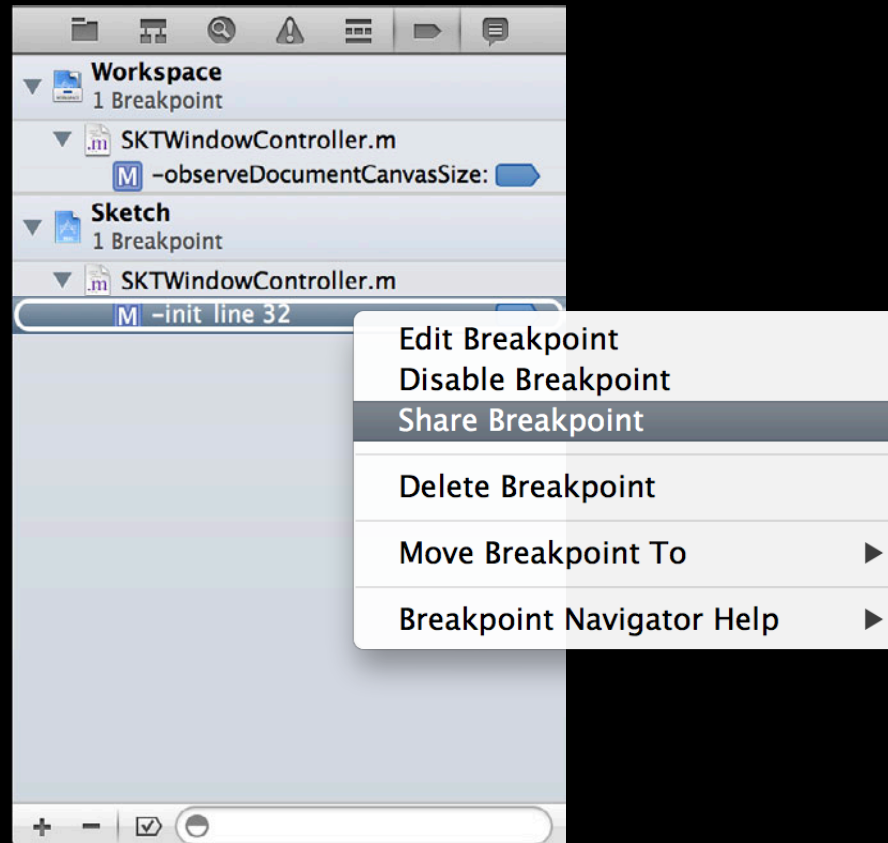
Managing Your Breakpoints

Project Breakpoint Group
Contains breakpoints that belong to the listed project

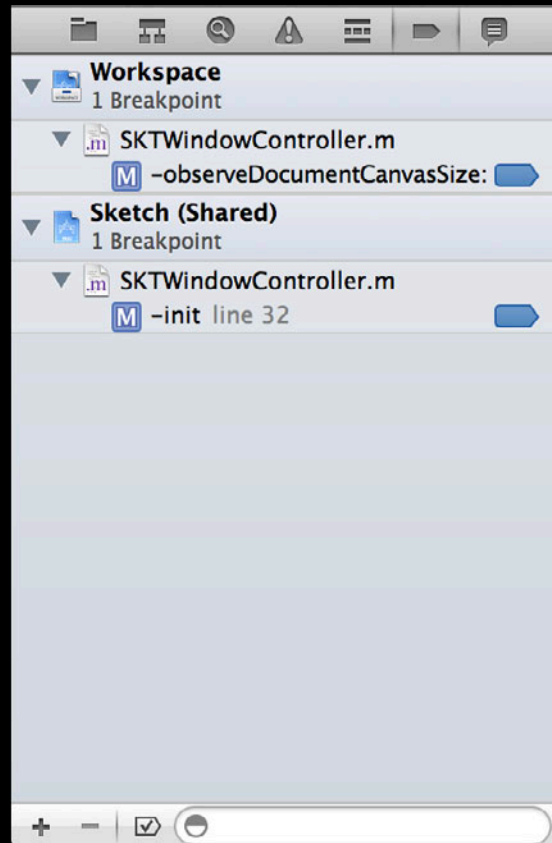


Managing Your Breakpoints

Project Breakpoint Group
Contains breakpoints that belong to the listed project

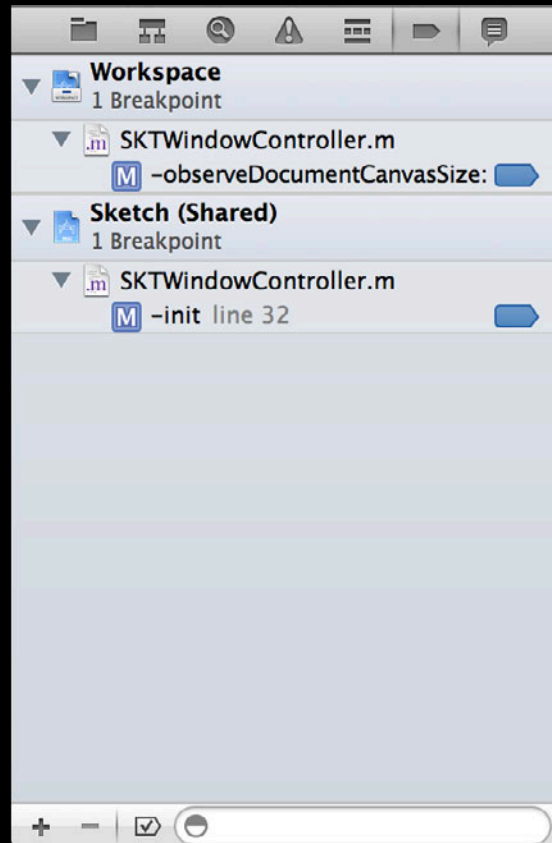


Managing Your Breakpoints

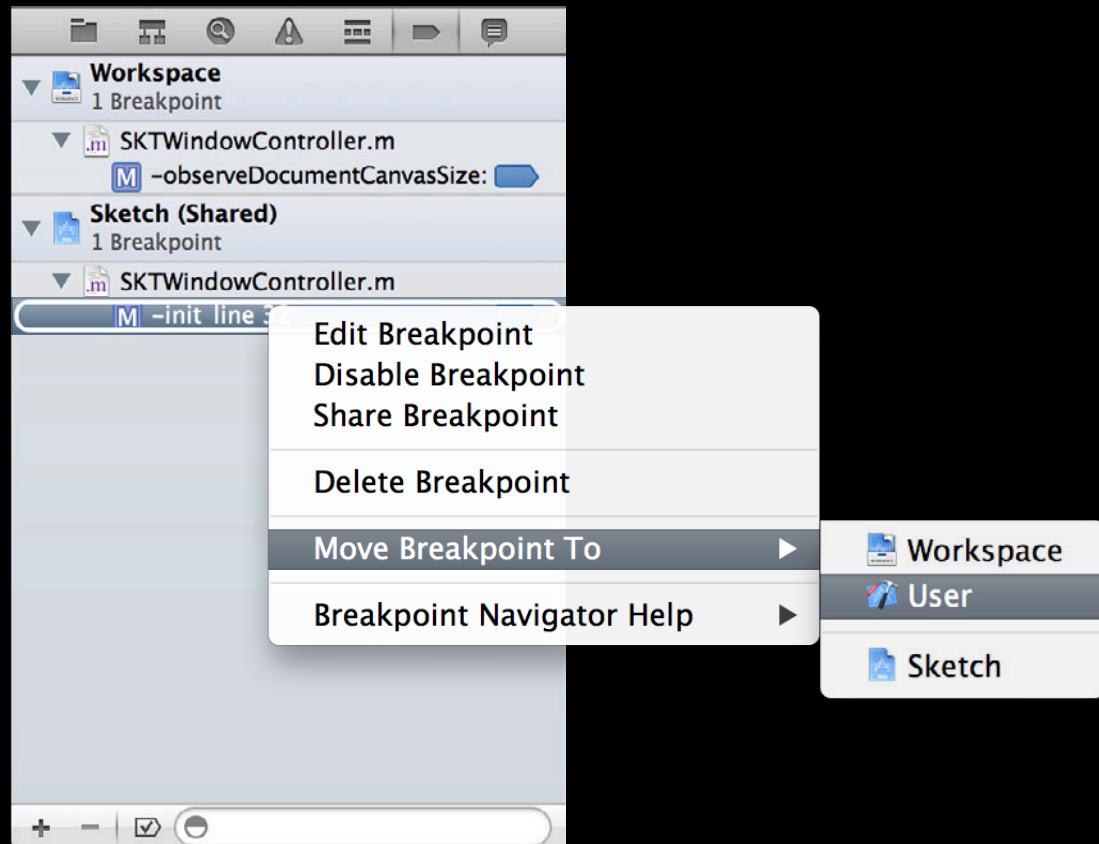


Managing Your Breakpoints

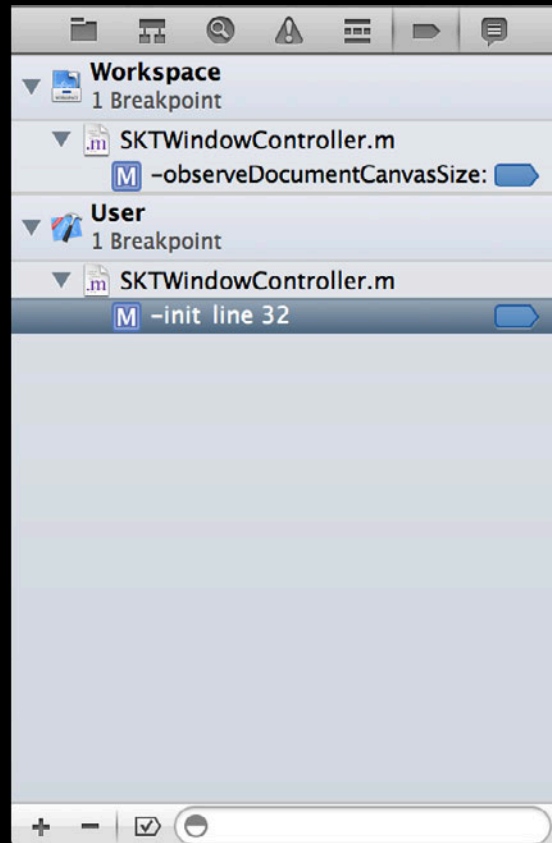
Shared Breakpoint Group
Shared breakpoints are available
to all users who open the project



Managing Your Breakpoints

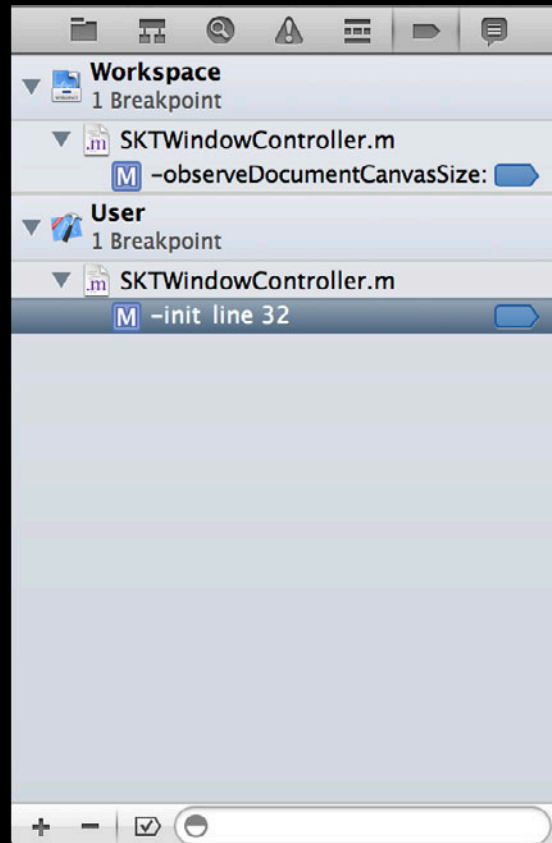


Managing Your Breakpoints

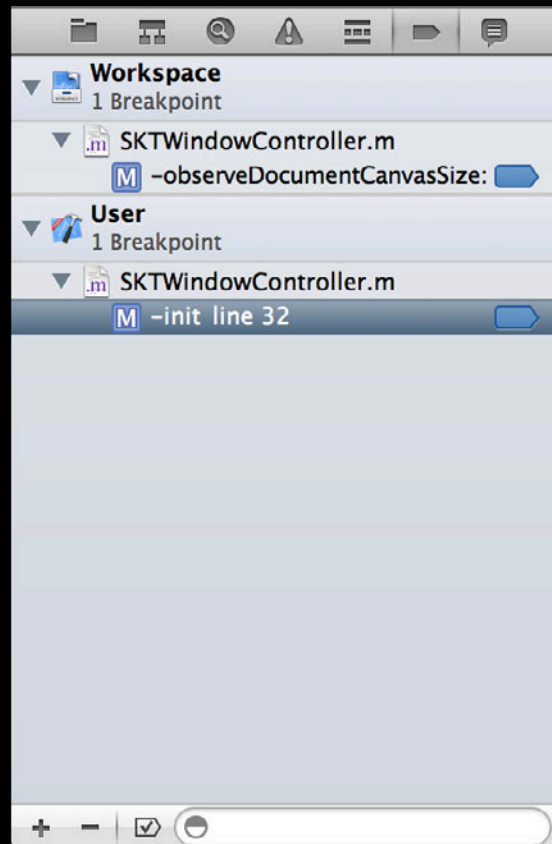


Managing Your Breakpoints

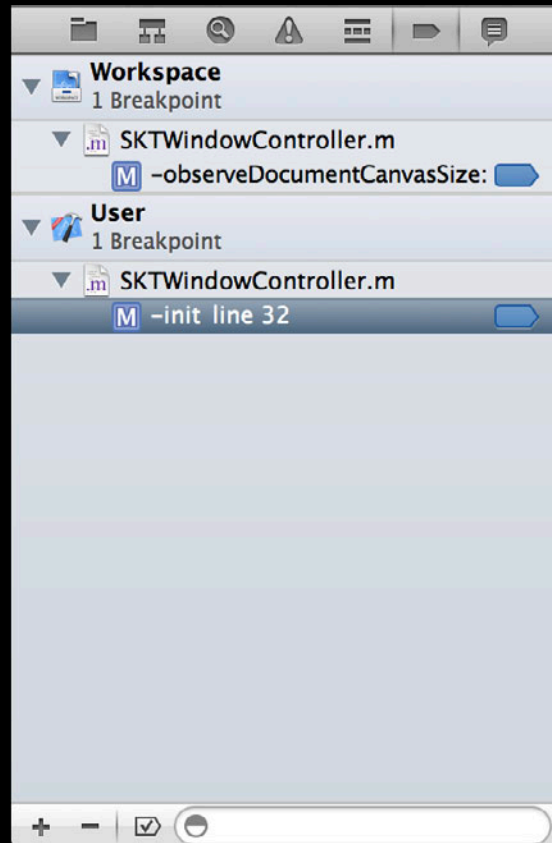
User Breakpoint Group
User breakpoints are available from all of your projects and workspaces



Exception and Symbolic Breakpoints



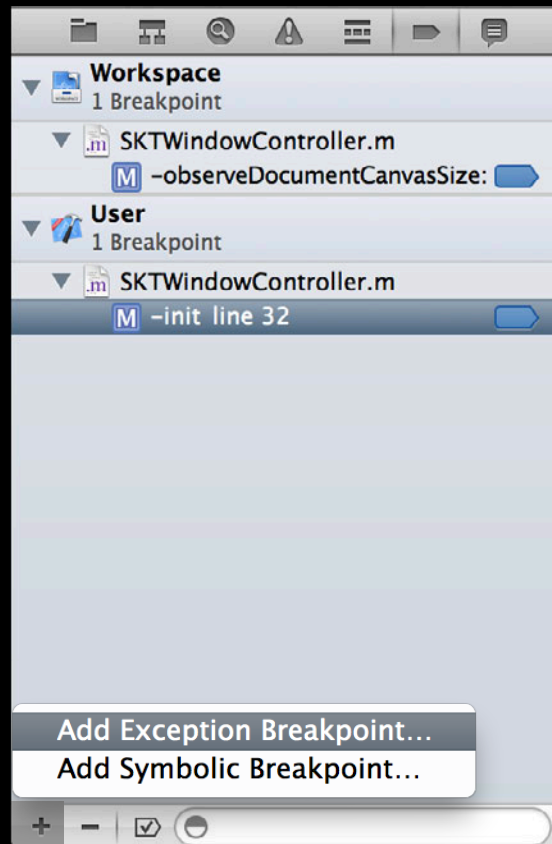
Exception and Symbolic Breakpoints



Add Breakpoint

Allows you to add an exception
or symbolic breakpoint

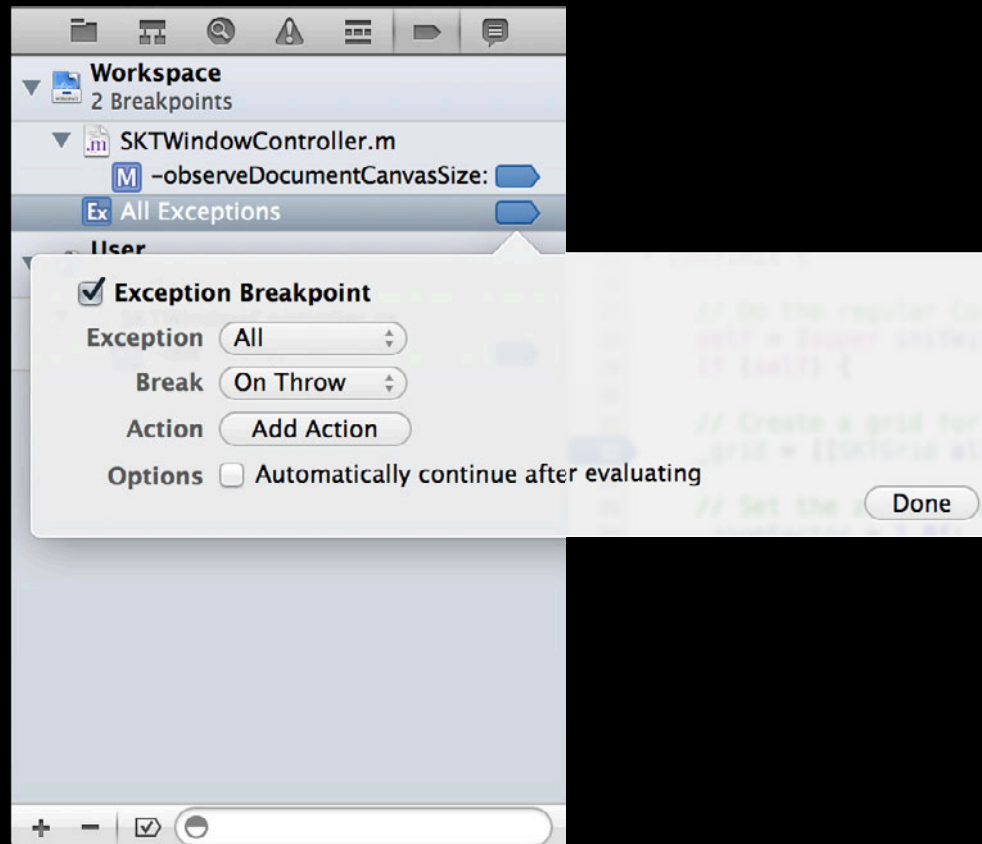
Exception and Symbolic Breakpoints



Add Breakpoint

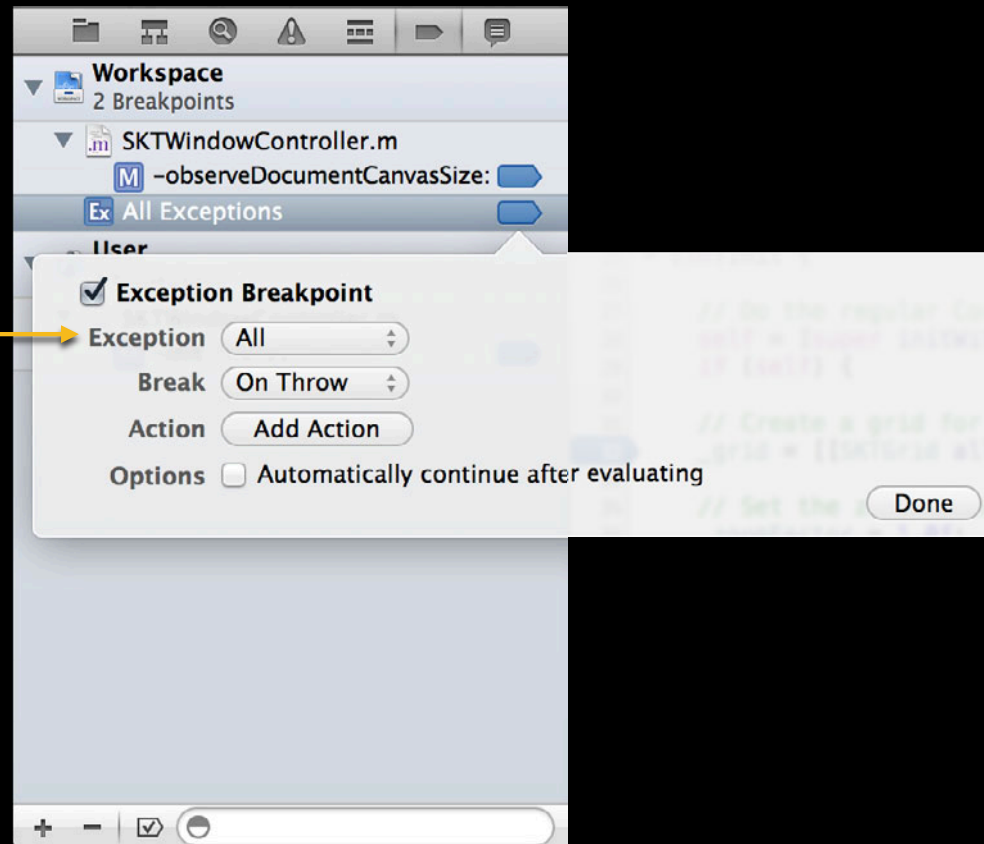
Allows you to add an exception
or symbolic breakpoint

Exception and Symbolic Breakpoints



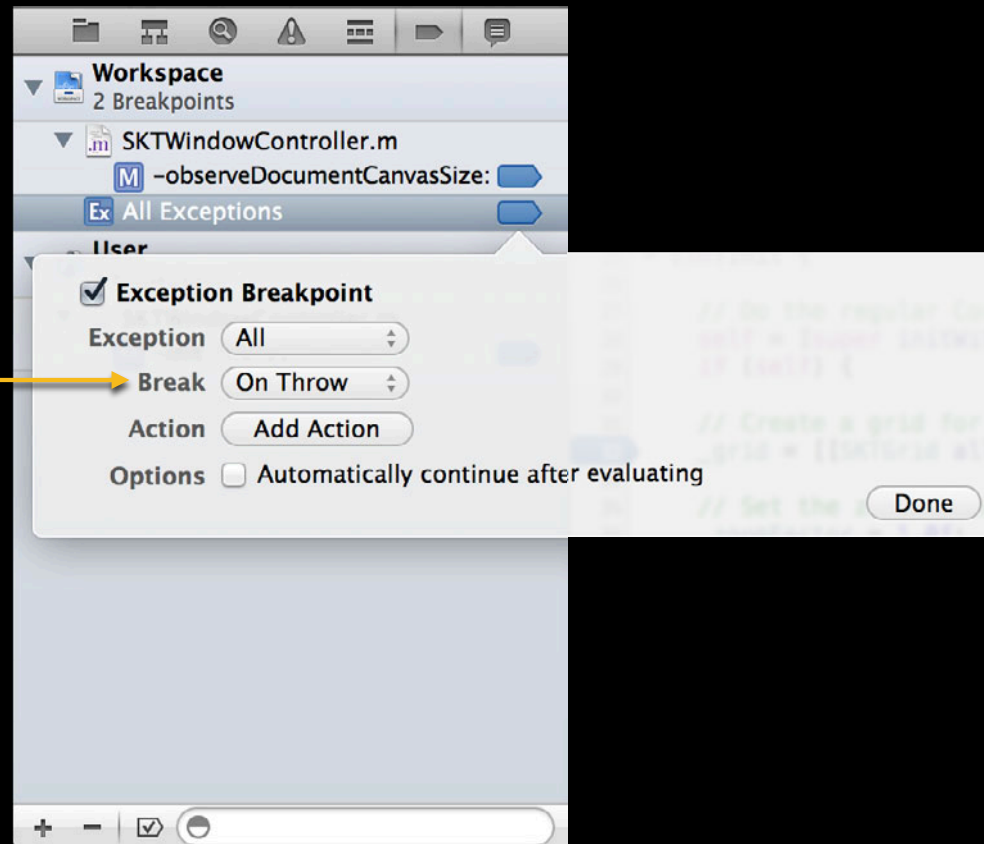
Exception and Symbolic Breakpoints

Type of Exception Breakpoint
Allows you to stop on all exceptions, Objective-C exceptions, or C++ exceptions

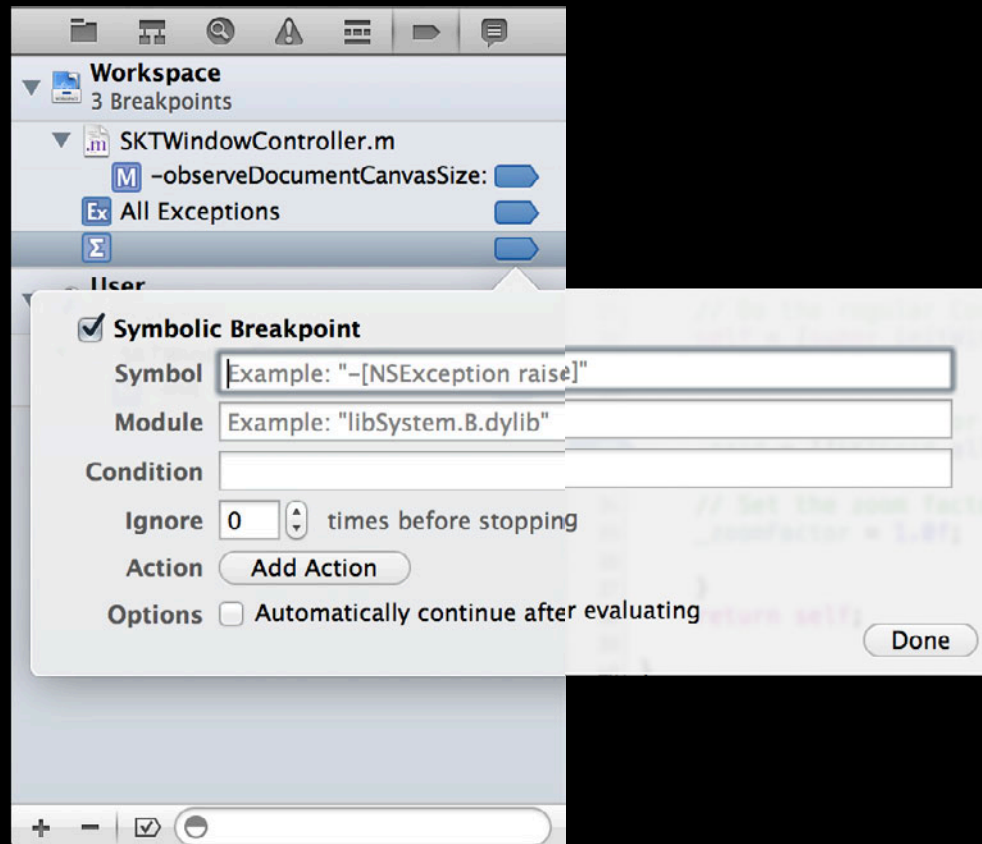


Exception and Symbolic Breakpoints

When to Break
Allows you to specify whether you want to break on catch or throw

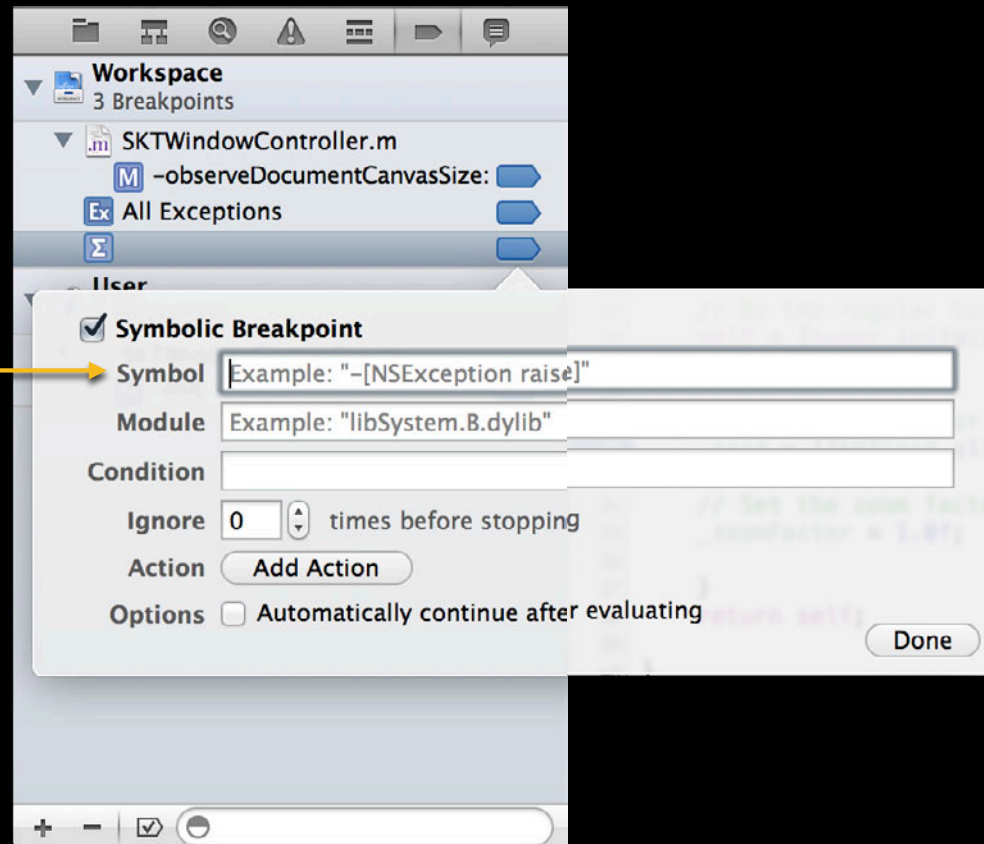


Exception and Symbolic Breakpoints

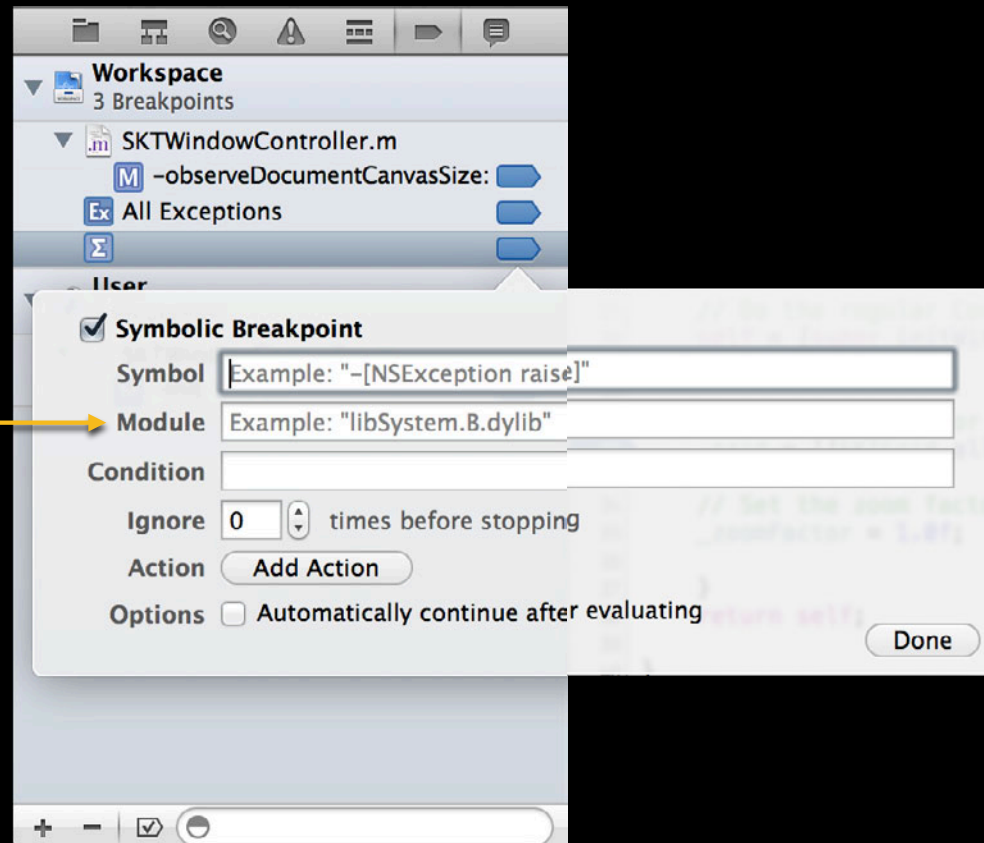


Exception and Symbolic Breakpoints

Symbol Name
Execution will pause when a
symbol with this name is called



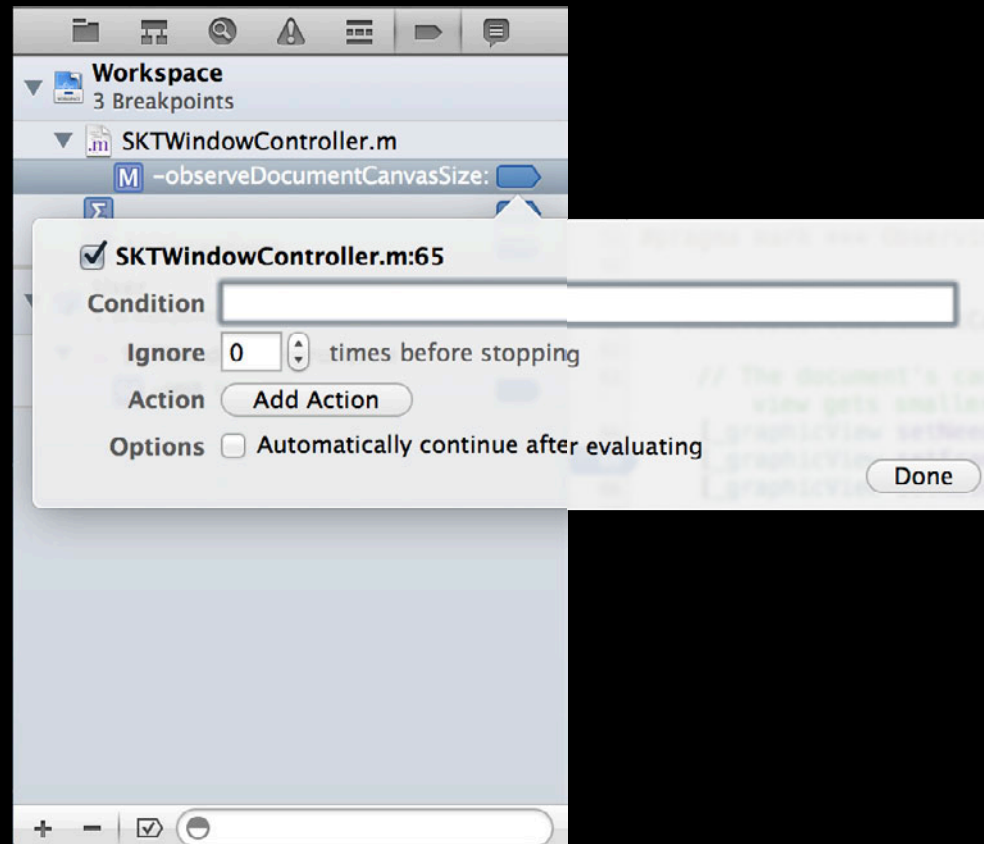
Exception and Symbolic Breakpoints



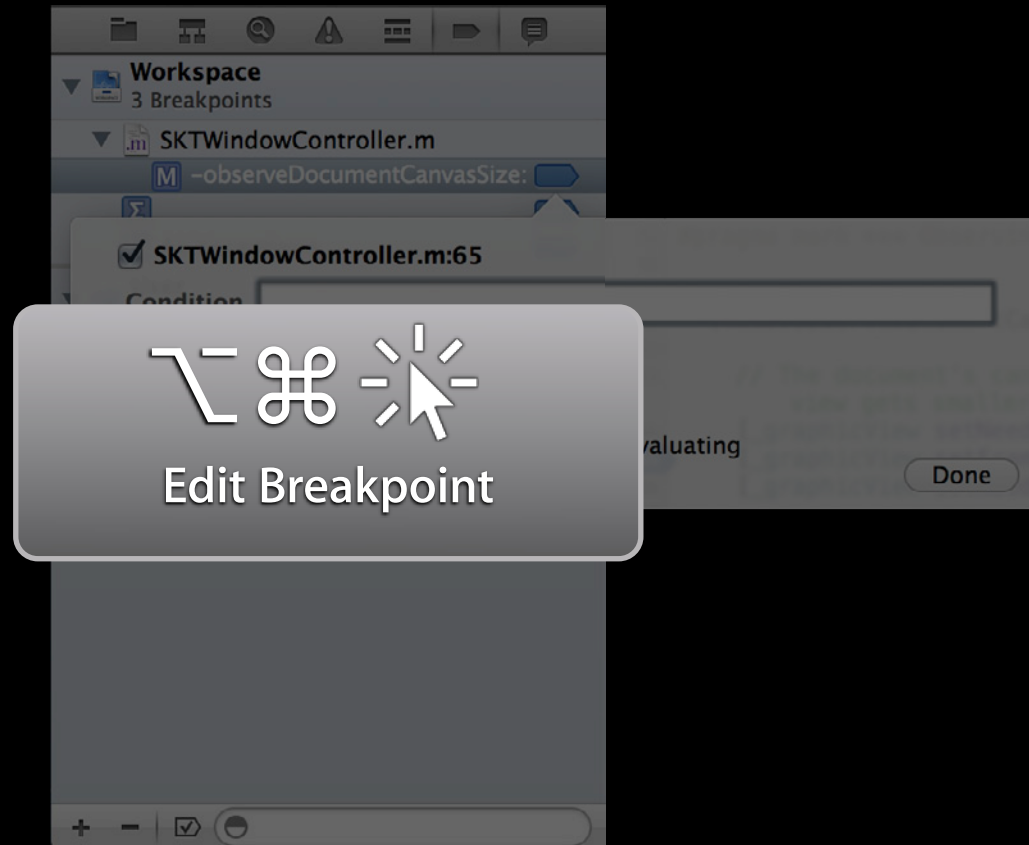
Module Name

Lets you restrict what libraries with the specified symbol name should result in a pause

Editing Breakpoints



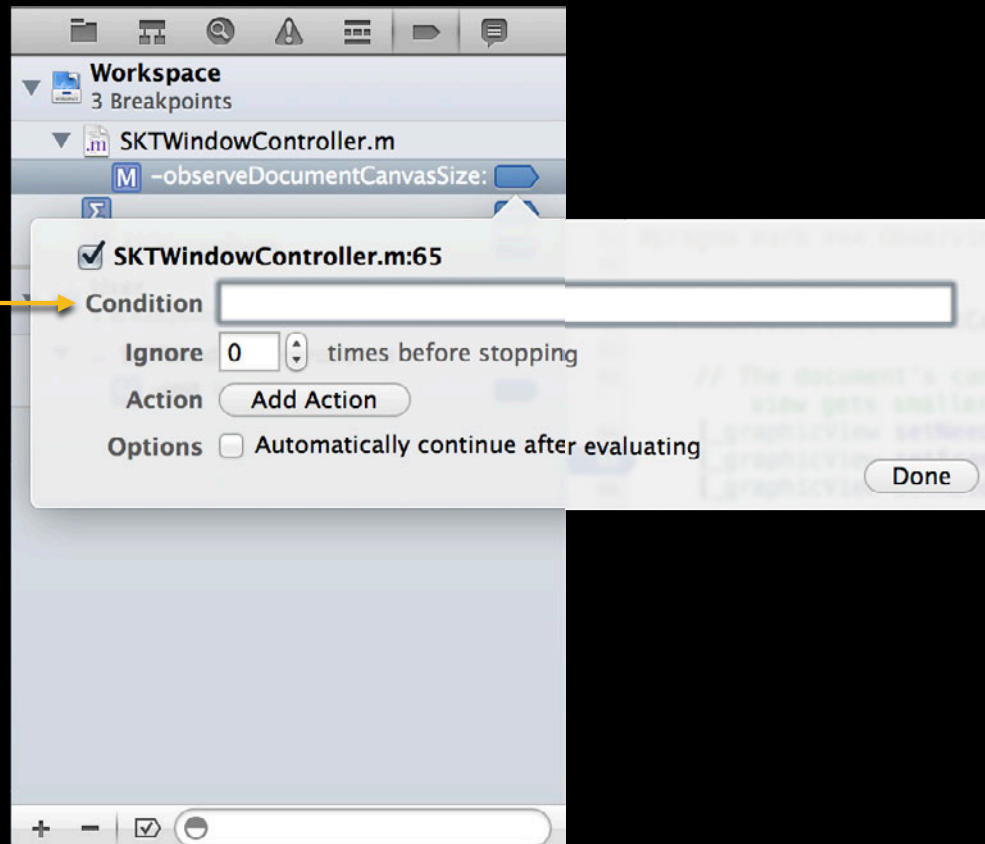
Editing Breakpoints



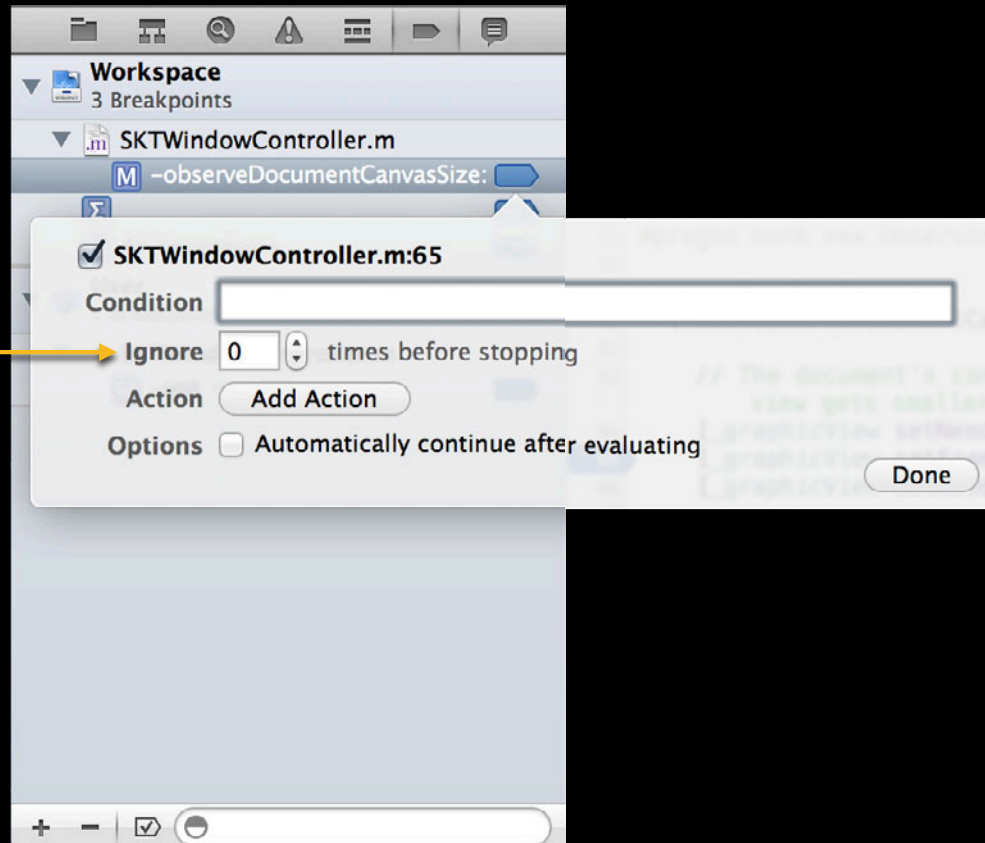
Editing Breakpoints

Condition to Evaluate

An expression to evaluate in order to determine if the breakpoint should be stopped at



Editing Breakpoints

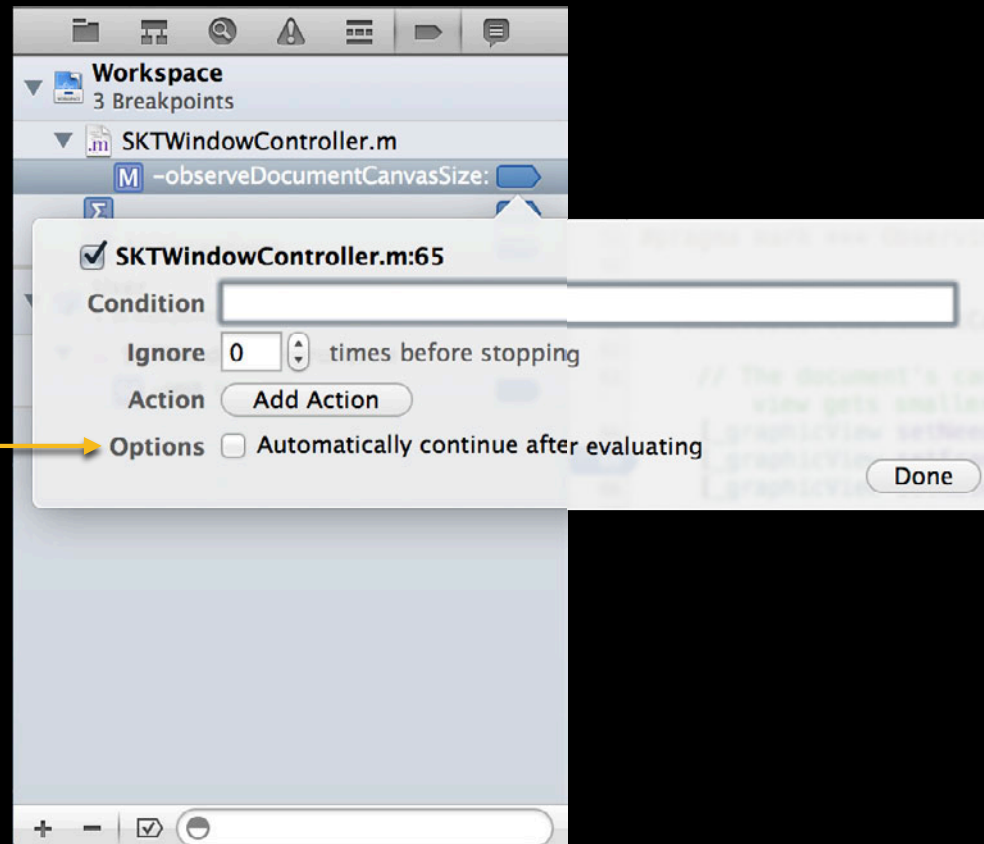


Ignore Count

The number of times to ignore the breakpoint before stopping

Editing Breakpoints

Automatically Continue
Continues program execution
after all of the breakpoint's
actions have completed



Breakpoint Actions

SKTWindowController.m:65

Condition

Ignore times before stopping

Action

Options Automatically continue after evaluating

Breakpoint Actions

SKTWindowController.m:65

Condition

Ignore times before stopping

Action

Options Automatically continue after evaluating

Breakpoint Actions

SKTWindowController.m:65

Condition

Ignore times before stopping

Action

Options Automatically continue after evaluating

Breakpoint Actions

SKTWindowController.m:65

Condition

Ignore times before stopping

Action + -

Options Automatically continue after evaluating

Done

Breakpoint Actions

SKTWindowController.m:65

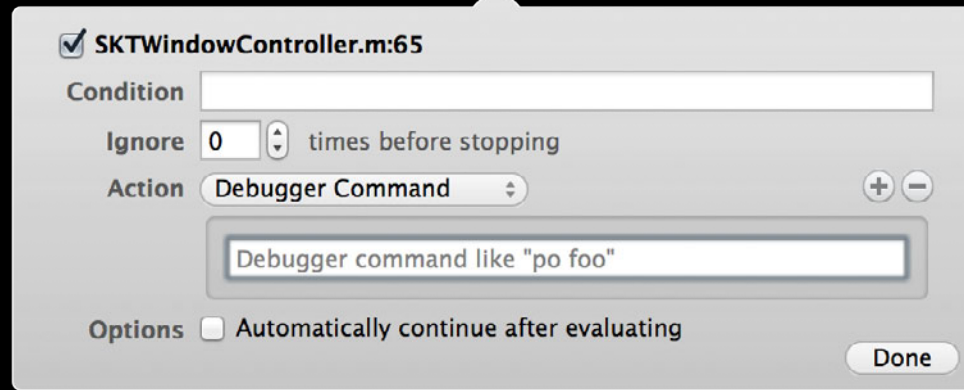
Condition

Ignore times before stopping

Action

Options Automatically continue after evaluating

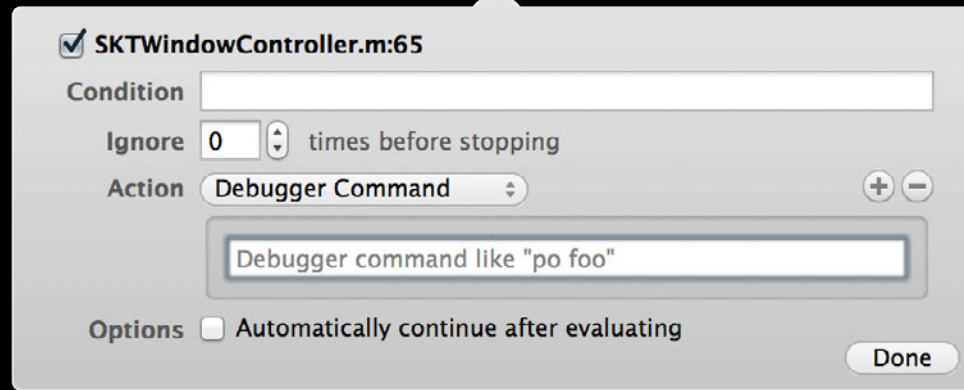
Debugger Command Breakpoint Action



A screenshot of a debugger breakpoint configuration dialog box. The dialog is titled "SKTWindowController.m:65" and has a checked checkbox next to it. It contains several fields and options:

- Condition:** An empty text input field.
- Ignore:** A numeric input field containing "0" with up and down arrow buttons, followed by the text "times before stopping".
- Action:** A dropdown menu currently showing "Debugger Command" with plus and minus buttons to its right.
- Command:** A text input field containing the text "Debugger command like 'po foo'".
- Options:** A checkbox labeled "Automatically continue after evaluating" which is currently unchecked.
- Done:** A button in the bottom right corner.

Debugger Command Breakpoint Action

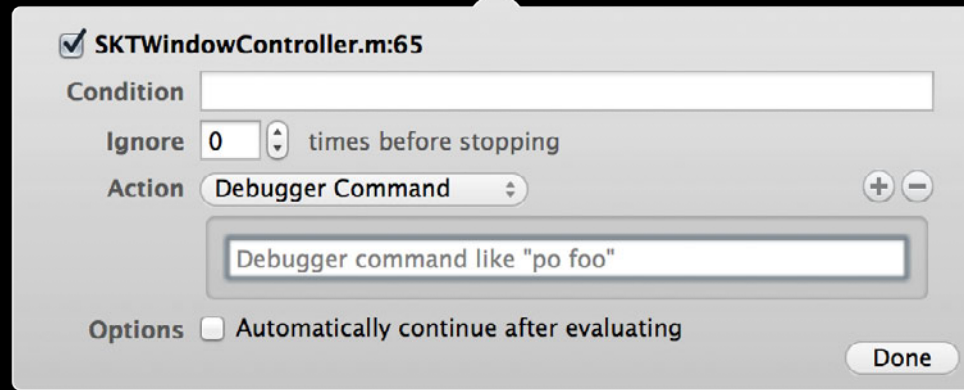


A screenshot of a debugger breakpoint configuration dialog box. The dialog is titled "SKTWindowController.m:65" and has a checked checkbox next to it. It contains the following fields and controls:

- Condition:** An empty text input field.
- Ignore:** A numeric input field containing "0" and a spinner control, followed by the text "times before stopping".
- Action:** A dropdown menu currently showing "Debugger Command" with plus and minus icons to its right.
- Command:** A text input field containing the text "Debugger command like 'po foo'".
- Options:** A checkbox labeled "Automatically continue after evaluating" which is currently unchecked.
- Done:** A button in the bottom right corner.

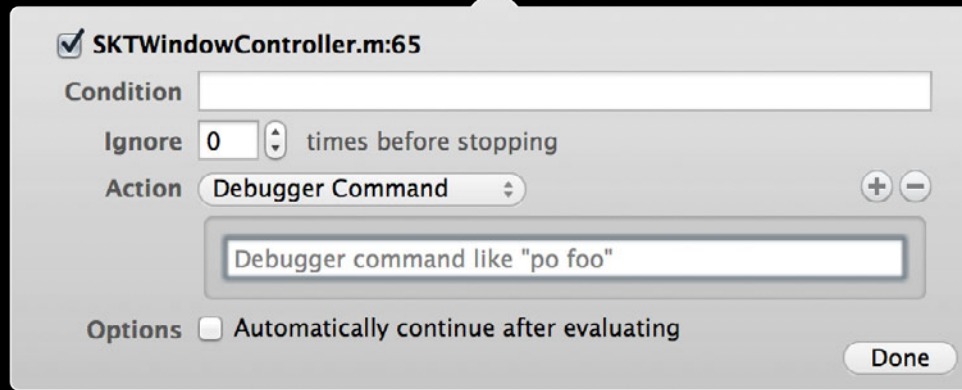
`po myVariable`

Debugger Command Breakpoint Action



```
po myVariable  
expr (void)NSLog(@"%f", myVariable)
```


Debugger Command Breakpoint Action



```
po myVariable
```

```
expr (void)NSLog(@"%f", myVariable)
```

```
breakpoint set -f SKTWindowController.m -l 100
```

Log Message Breakpoint Action

✓ SKTWindowController.m:65

Condition

Ignore times before stopping

Action + -

Log message to console @exp@ = expression
 Speak message %B = breakpoint name
%H = breakpoint hit count

Options Automatically continue after evaluating

Done

Log Message Breakpoint Action

The screenshot shows the configuration for a breakpoint action in Xcode. The breakpoint is named "SKTWindowController.m:65" and is checked. The "Condition" field is empty. The "Ignore" field is set to "0" times before stopping. The "Action" is set to "Log Message". The "Message" field contains the text "Message". The "Log message to console" option is selected, and the "Speak message" option is unselected. The "Options" section has the "Automatically continue after evaluating" checkbox unchecked. A "Done" button is located at the bottom right.

✓ SKTWindowController.m:65

Condition

Ignore times before stopping

Action + -

Log message to console @exp@ = expression
 Speak message %B = breakpoint name
%H = breakpoint hit count

Options Automatically continue after evaluating

Done

`myVariable = @myVariable@, hit %H times`

Shell Command Breakpoint Action

SKTWindowController.m:65

Condition

Ignore times before stopping

Action

Wait until done @exp@ = expression
%B = breakpoint name
%H = breakpoint hit count

Options Automatically continue after evaluating

Shell Command Breakpoint Action

✓ SKTWindowController.m:65

Condition

Ignore times before stopping

Action

Wait until done @exp@ = expression
%B = breakpoint name
%H = breakpoint hit count

Options Automatically continue after evaluating

Command: **screencapture**

Arguments: **/tmp/screenShot.png**

Sound Breakpoint Action

SKTWindowController.m:65

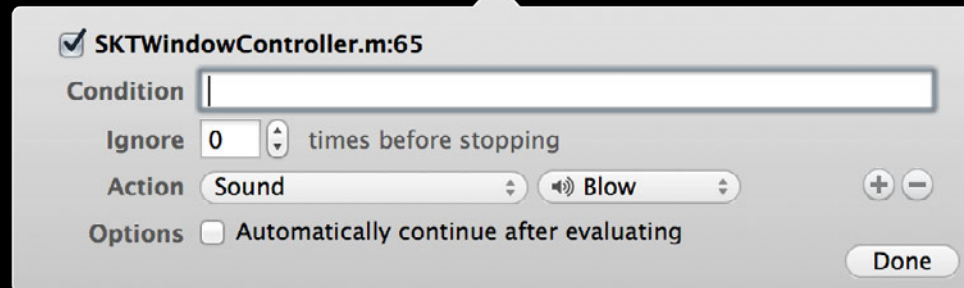
Condition

Ignore times before stopping

Action

Options Automatically continue after evaluating

Sound Breakpoint Action



~/Library/Sounds
Sounds put in your home
directory show up in Xcode



Data > Users > korr > Library > Sounds

AppleScript Breakpoint Action

SKTWindowController.m:65

Condition

Ignore times before stopping

Action

@exp@ = expression
%B = breakpoint name
%H = breakpoint hit count

Press compile to verify your script

Options Automatically continue after evaluating

AppleScript Breakpoint Action

```
tell application "Mail"  
  set myMessage to make new outgoing message with properties  
    {visible:false, subject:"Test Failed", content:"The test failed"}  
  tell myMessage  
    make new to recipient at end of to recipients with properties  
      {name:"Ken Orr", address:"orr@apple.com"}  
    myMessage send  
  end tell  
end tell
```

Demo

Breakpoints and breakpoint actions

Alex Raftis

Xcode Debugger UI Engineer

The Variables View

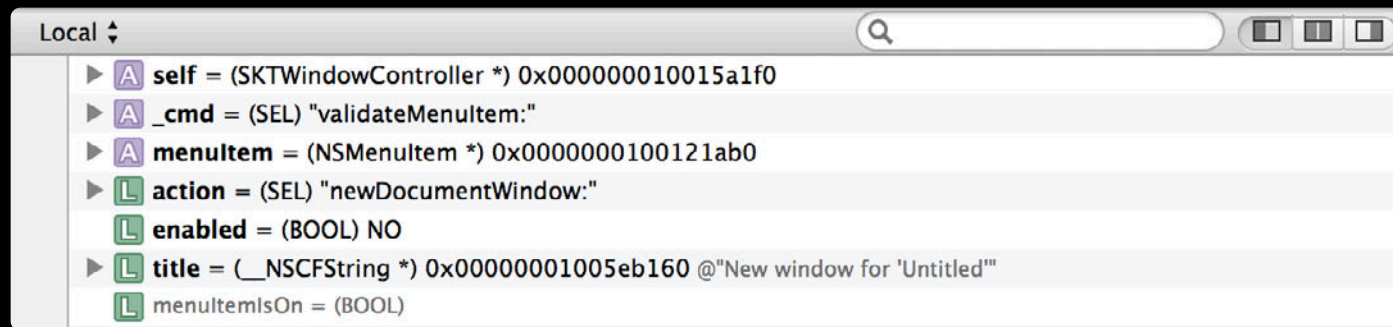
Viewing Your Variables

The screenshot shows the Xcode IDE with a breakpoint hit in the file `SKTWindowController.m`. The code is paused at line 179, which is `enabled = YES;`. The Local variables pane shows the following variables:

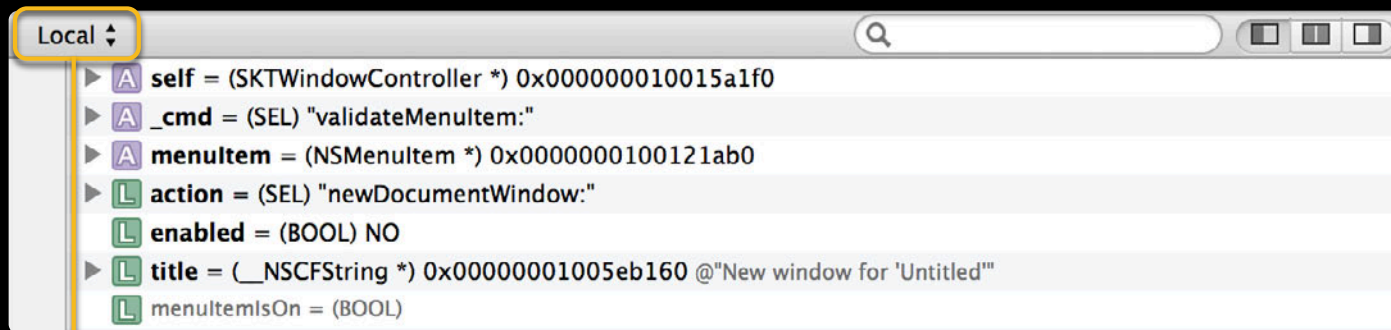
- `self` = (SKTWindowController *) 0x000000010015a1f0
- `_cmd` = (SEL) "validateMenuItem:"
- `menuItem` = (NSMenuItem *) 0x0000000100121ab0
- `action` = (SEL) "newDocumentWindow:"
- `enabled` = (BOOL) NO
- `title` = (__NSCFString *) 0x00000001005eb160 @"New window for 'Untitled'"
- `menuItemIsOn` = (BOOL)

```
descriptive title. It's important to use the document's "display name" in places like
this; it takes things like file name extension hiding into account. We could do a
better job with the punctuation!
177 NSString *title = [NSString stringWithFormat:NSLocalizedStringFromTable(@"New window for
'%@'", @"MenuItems", @"Formatter string for the new document window menu item.
Argument is a the display name of the document."), [[self document] displayName]];
178 [menuItem setTitle:title];
179 enabled = YES;
180
181 } else if (action==@selector(toggleGridConstraining:) || action==@selector
(toggleGridShowing:)) {
182
183 // The grid can be in an unusable state, in which case the menu items that control it are
disabled.
184 enabled = [_grid isUsable];
185
```

Viewing Your Variables



Viewing Your Variables



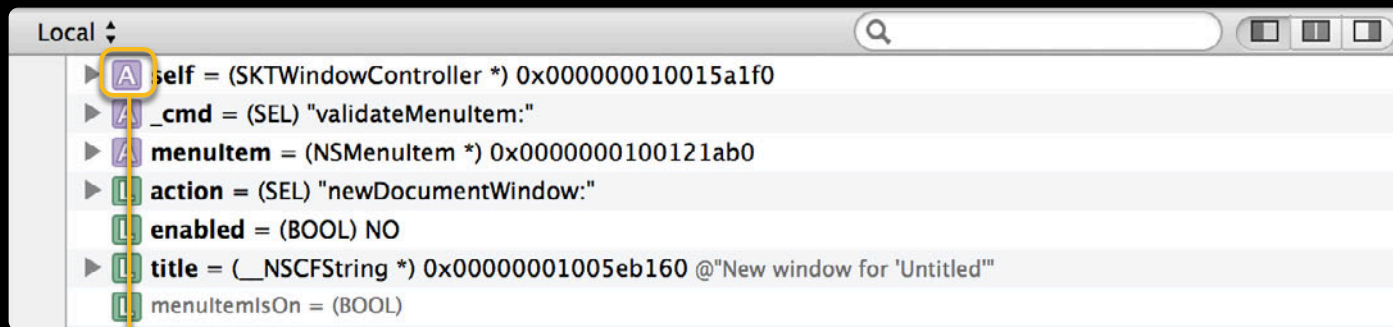
Viewing Mode

Auto - variables around the line of code you're paused at

Local - all variables in local scope

All - all variables including globals and registers

Viewing Your Variables



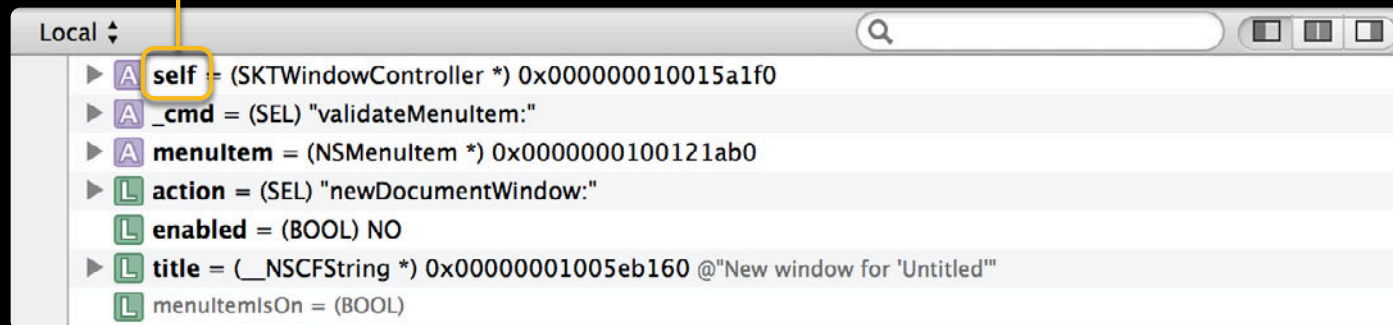
Variable Kind

- | | |
|--------------------------|----------------------------|
| L Local Variable | R Register |
| A Argument | V Instance Variable |
| S Static Variable | E Expression |
| V Global Variable | |

Viewing Your Variables

Variable Name

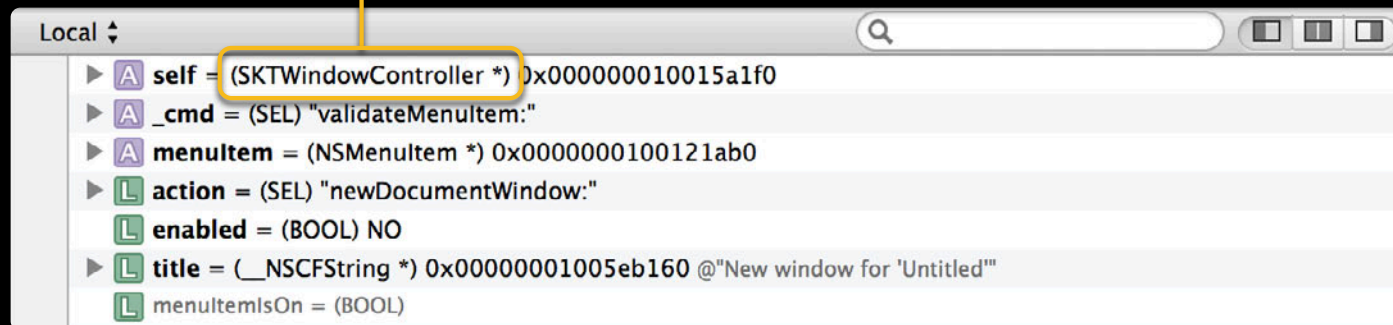
The name of the variable as it appears in your code



Viewing Your Variables

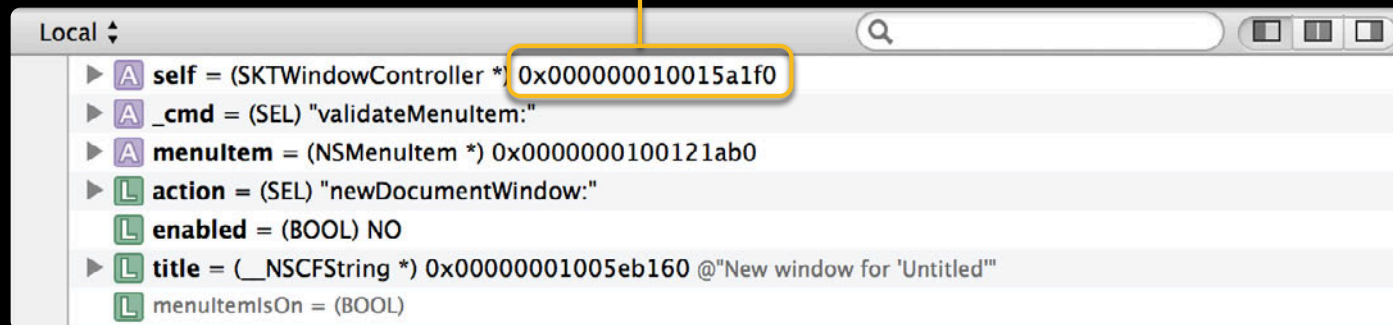
Dynamic Type

Runtime type of the variable

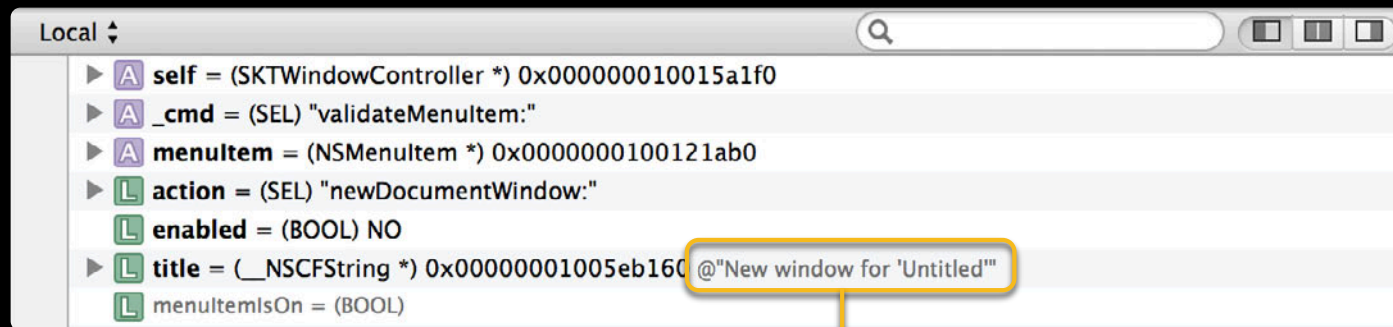


Viewing Your Variables

Variable Value
Current value of the variable



Viewing Your Variables



Variable Summary

Summary for the variable, if there is one

Return Value When Stepping Out



```
- (void)updateLabel
{
    [_textField setStringValue:[self currentProcessNameString]];
}

- (NSString *)currentProcessNameString
{
    return [NSString stringWithFormat:@"Current process is %@",
        [[NSProcessInfo processInfo] processName]];
}
```

Return Value When Stepping Out



```
- (void)updateLabel
{
    [_textField setStringValue:[self currentProcessNameString]];
}

- (NSString *)currentProcessNameString
{
    return [NSString stringWithFormat:@"Current process is %@",
        [[NSProcessInfo processInfo] processName]];
}
```

Return Value When Stepping Out



```
- (void)updateLabel
{
    [_textField setStringValue:[self currentProcessNameString]];
}

- (NSString *)currentProcessNameString
{
    return [NSString stringWithFormat:@"Current process is %@",
        [[NSProcessInfo processInfo] processName]];
}
```

Return Value When Stepping Out



```
- (void)updateLabel
{
    [_textField setStringValue:[self currentProcessNameString]];
}

- (NSString *)currentProcessNameString
{
    return [NSString stringWithFormat:@"Current process is %@",
        [[NSProcessInfo processInfo] processName]];
}
```

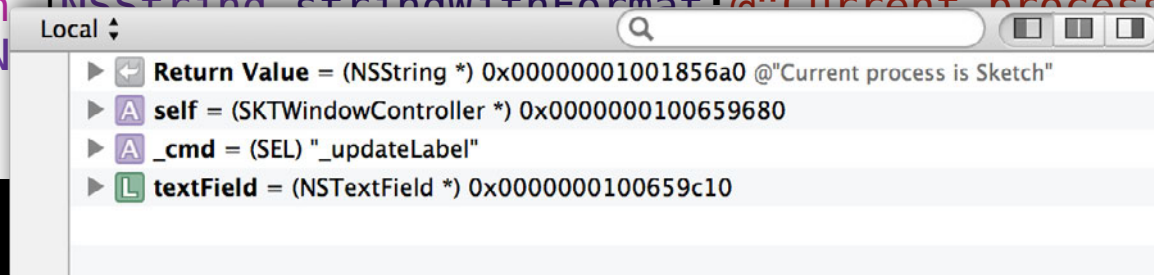


Return Value When Stepping Out



```
- (void)updateLabel
{
    [_textField setStringValue:[self currentProcessNameString]];
}

- (NSString *)currentProcessNameString
{
    return [NSString stringWithFormat:@"Current process is %@",
    [NSProcessInfo processInfo].processName];
}
```



Return Value When Stepping Out




```
- (void)updateLabel
{
    [_textField setStringValue:[self currentProcessNameString]];
}

- (NSString *)currentProcessNameString
{
    return [NSString stringWithFormat:@"Current process is %@",
        [NSProcessInfo processInfo].processName];
}
```


Local

- Return Value = (NSString *) 0x00000001001856a0 @"Current process is Sketch"
- self = (SKTWindowController *) 0x00000000100659680
- _cmd = (SEL) "_updateLabel"
- textField = (NSTextField *) 0x00000000100659c10

A Bit About Variable Summaries

 `title = (__NSCFString *) 0x1234 @"WWDC12"`

A Bit About Variable Summaries



```
title = (__NSCFString *) 0x1234 @"WWDC12"
```

The code snippet shows a variable declaration for 'title'. The value is a pointer to an NSString object, represented as '0x1234 @"WWDC12"'. A yellow underline is drawn under the string literal '"WWDC12"', with a vertical line extending downwards from its center.

Previously implemented with expressions: `[title description]`

A Bit About Variable Summaries

A Bit About Variable Summaries

A Bit About Variable Summaries



Can change the state
of your program

A Bit About Variable Summaries



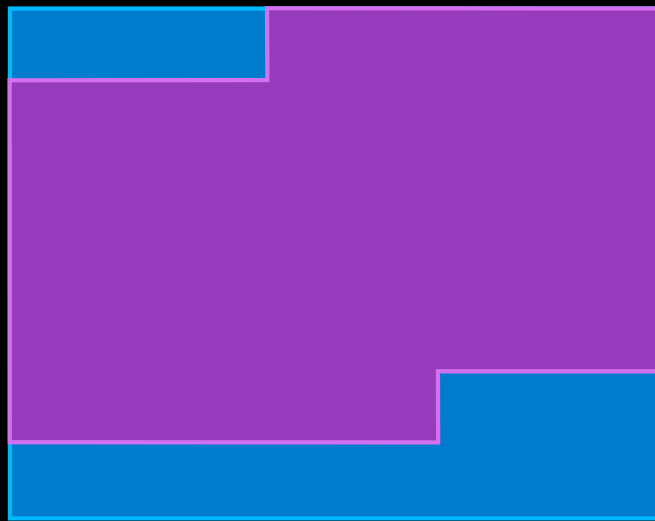
Can change the state
of your program



May not always work

A Bit About Variable Summaries

LLDB



Block of memory representing
`__NSCFString`

A Bit About Variable Summaries

LLDB

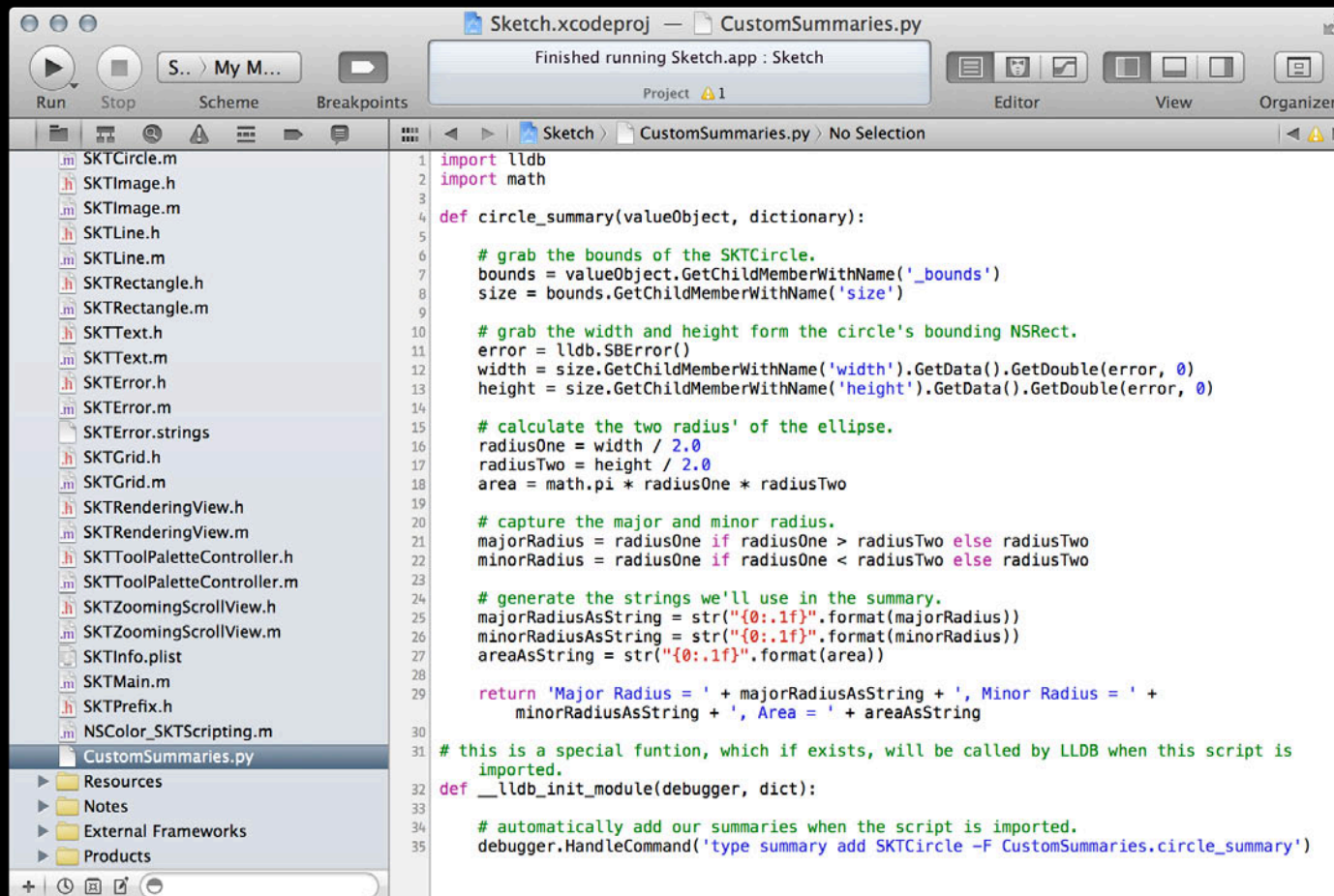


← **Character Data**
LLDB looks inside the object
and reads the memory of the
specific data it wants

Block of memory representing
_NSCFString

Creating Custom Summaries

LLDB

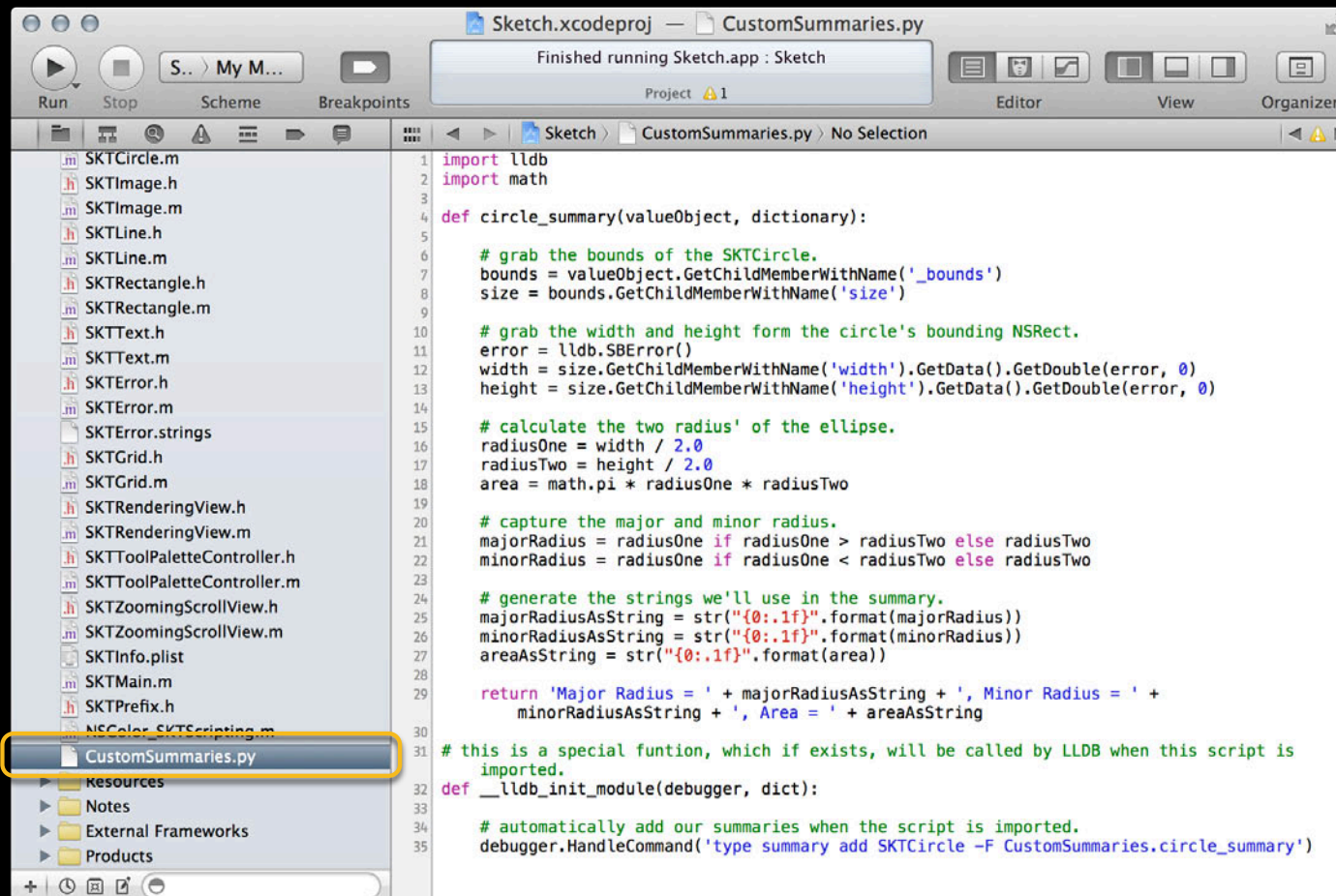


The screenshot shows the Xcode IDE with a project named 'Sketch.xcodeproj' and a file named 'CustomSummaries.py'. The interface includes a toolbar with 'Run', 'Stop', 'Scheme', and 'Breakpoints' buttons. A status bar at the top indicates 'Finished running Sketch.app : Sketch' and 'Project 1'. The left sidebar shows a file explorer with various source files, including 'SKTCircle.m' and 'CustomSummaries.py'. The main editor displays the following Python code:

```
1 import lldb
2 import math
3
4 def circle_summary(valueObject, dictionary):
5
6     # grab the bounds of the SKTCircle.
7     bounds = valueObject.GetChildMemberWithName('_bounds')
8     size = bounds.GetChildMemberWithName('size')
9
10    # grab the width and height form the circle's bounding NSRect.
11    error = lldb.SBError()
12    width = size.GetChildMemberWithName('width').GetData().GetDouble(error, 0)
13    height = size.GetChildMemberWithName('height').GetData().GetDouble(error, 0)
14
15    # calculate the two radius' of the ellipse.
16    radiusOne = width / 2.0
17    radiusTwo = height / 2.0
18    area = math.pi * radiusOne * radiusTwo
19
20    # capture the major and minor radius.
21    majorRadius = radiusOne if radiusOne > radiusTwo else radiusTwo
22    minorRadius = radiusOne if radiusOne < radiusTwo else radiusTwo
23
24    # generate the strings we'll use in the summary.
25    majorRadiusAsString = str("{0:.1f}".format(majorRadius))
26    minorRadiusAsString = str("{0:.1f}".format(minorRadius))
27    areaAsString = str("{0:.1f}".format(area))
28
29    return 'Major Radius = ' + majorRadiusAsString + ', Minor Radius = ' +
30        minorRadiusAsString + ', Area = ' + areaAsString
31
32 # this is a special funtion, which if exists, will be called by LLDB when this script is
33 # imported.
34 def __lldb_init_module(debugger, dict):
35
36     # automatically add our summaries when the script is imported.
37     debugger.HandleCommand('type summary add SKTCircle -F CustomSummaries.circle_summary')
```

Creating Custom Summaries

LLDB

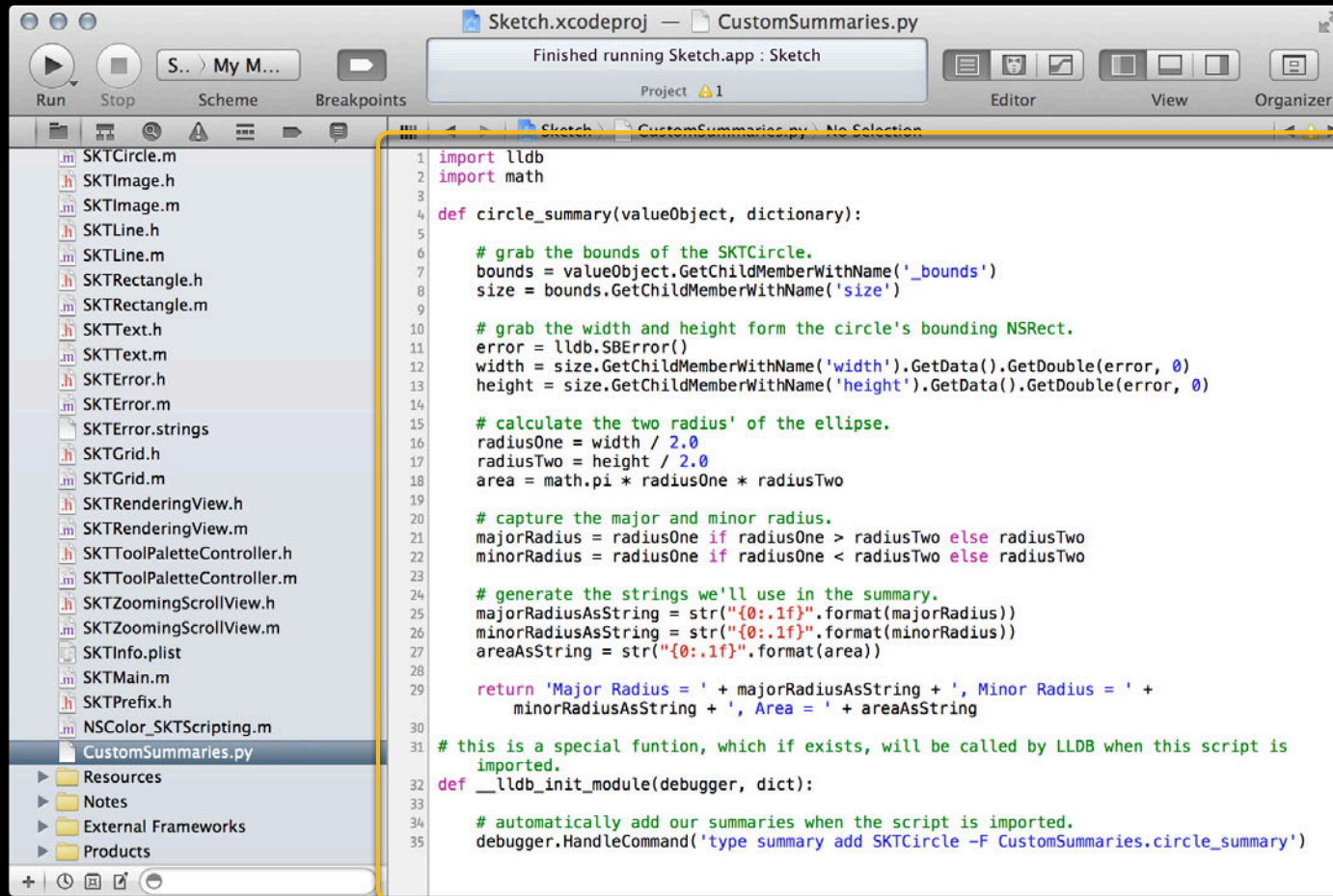


The screenshot shows the Xcode IDE with a Python script named CustomSummaries.py. The script defines a function circle_summary and an LLDB initialization function __lldb_init_module. The circle_summary function calculates the major and minor radii and the area of a circle based on its bounding box. The __lldb_init_module function registers the circle_summary function with LLDB for the SKTCircle class.

```
1 import lldb
2 import math
3
4 def circle_summary(valueObject, dictionary):
5
6     # grab the bounds of the SKTCircle.
7     bounds = valueObject.GetChildMemberWithName('_bounds')
8     size = bounds.GetChildMemberWithName('size')
9
10    # grab the width and height form the circle's bounding NSRect.
11    error = lldb.SBError()
12    width = size.GetChildMemberWithName('width').GetData().GetDouble(error, 0)
13    height = size.GetChildMemberWithName('height').GetData().GetDouble(error, 0)
14
15    # calculate the two radius' of the ellipse.
16    radiusOne = width / 2.0
17    radiusTwo = height / 2.0
18    area = math.pi * radiusOne * radiusTwo
19
20    # capture the major and minor radius.
21    majorRadius = radiusOne if radiusOne > radiusTwo else radiusTwo
22    minorRadius = radiusOne if radiusOne < radiusTwo else radiusTwo
23
24    # generate the strings we'll use in the summary.
25    majorRadiusAsString = str("{0:.1f}".format(majorRadius))
26    minorRadiusAsString = str("{0:.1f}".format(minorRadius))
27    areaAsString = str("{0:.1f}".format(area))
28
29    return 'Major Radius = ' + majorRadiusAsString + ', Minor Radius = ' +
30        minorRadiusAsString + ', Area = ' + areaAsString
31
32 # this is a special funtion, which if exists, will be called by LLDB when this script is
33 # imported.
34 def __lldb_init_module(debugger, dict):
35
36    # automatically add our summaries when the script is imported.
37    debugger.HandleCommand('type summary add SKTCircle -F CustomSummaries.circle_summary')
```

Creating Custom Summaries

LLDB



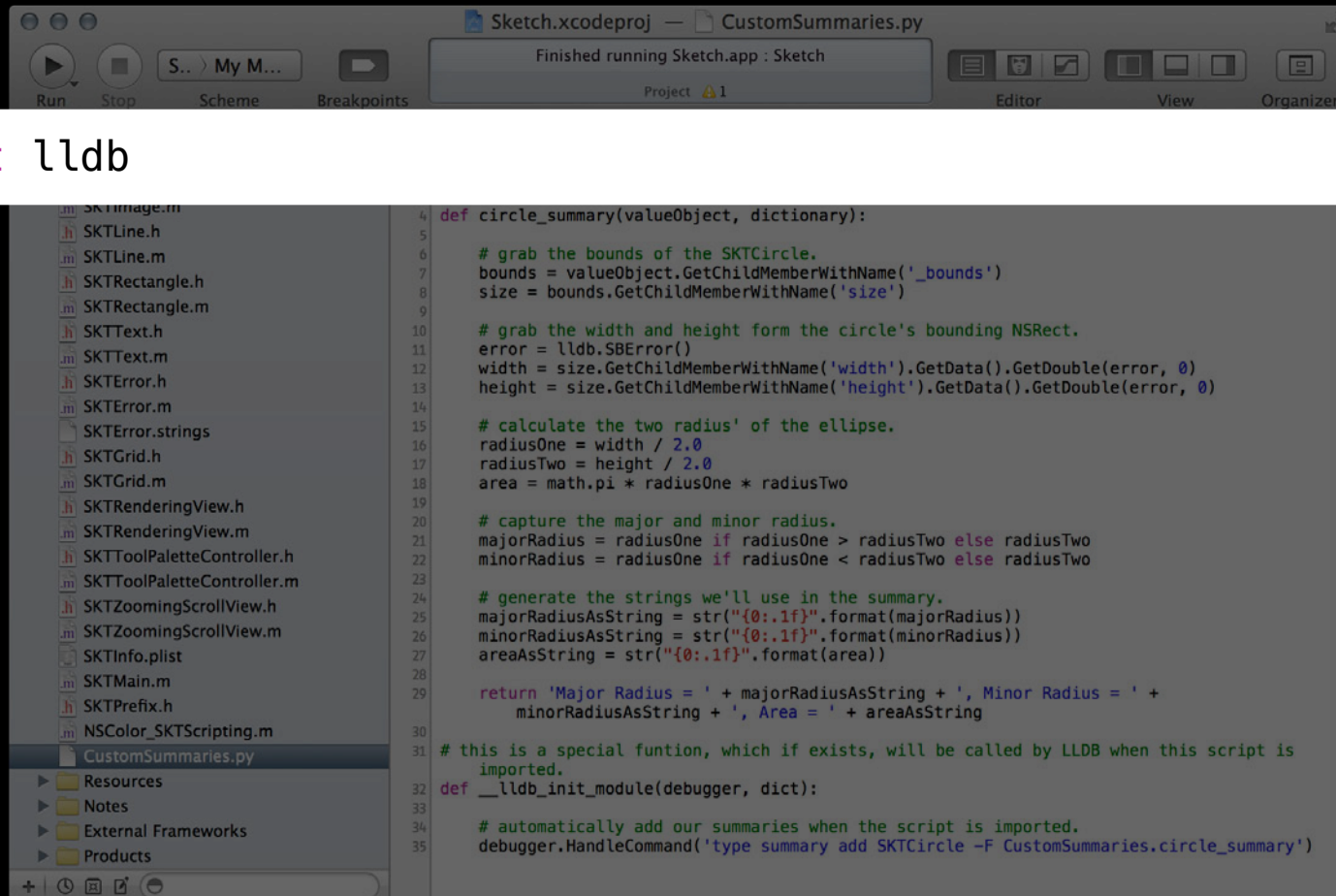
The screenshot shows the Xcode IDE with a Python script named `CustomSummaries.py` open in the editor. The script defines a custom summary function for `SKTCircle` objects. The function, `circle_summary`, takes a `valueObject` and a `dictionary` as input. It uses LLDB's `GetChildMemberWithName` to retrieve the `_bounds` member, which contains a `size` member representing the bounding rectangle. It then calculates the major and minor radii from the width and height of the bounding rectangle and computes the area. The function returns a string summarizing these values: "Major Radius = [majorRadius], Minor Radius = [minorRadius], Area = [area]". Additionally, the script includes a special function `__lldb_init_module` that registers the custom summary with LLDB using `debugger.HandleCommand('type summary add SKTCircle -F CustomSummaries.circle_summary')`.

```
1 import lldb
2 import math
3
4 def circle_summary(valueObject, dictionary):
5
6     # grab the bounds of the SKTCircle.
7     bounds = valueObject.GetChildMemberWithName('_bounds')
8     size = bounds.GetChildMemberWithName('size')
9
10    # grab the width and height form the circle's bounding NSRect.
11    error = lldb.SBError()
12    width = size.GetChildMemberWithName('width').GetData().GetDouble(error, 0)
13    height = size.GetChildMemberWithName('height').GetData().GetDouble(error, 0)
14
15    # calculate the two radius' of the ellipse.
16    radiusOne = width / 2.0
17    radiusTwo = height / 2.0
18    area = math.pi * radiusOne * radiusTwo
19
20    # capture the major and minor radius.
21    majorRadius = radiusOne if radiusOne > radiusTwo else radiusTwo
22    minorRadius = radiusOne if radiusOne < radiusTwo else radiusTwo
23
24    # generate the strings we'll use in the summary.
25    majorRadiusAsString = str("{0:.1f}".format(majorRadius))
26    minorRadiusAsString = str("{0:.1f}".format(minorRadius))
27    areaAsString = str("{0:.1f}".format(area))
28
29    return 'Major Radius = ' + majorRadiusAsString + ', Minor Radius = ' +
30        minorRadiusAsString + ', Area = ' + areaAsString
31
32 # this is a special funtion, which if exists, will be called by LLDB when this script is
33 # imported.
34 def __lldb_init_module(debugger, dict):
35
36     # automatically add our summaries when the script is imported.
37     debugger.HandleCommand('type summary add SKTCircle -F CustomSummaries.circle_summary')
```

Creating Custom Summaries

LLDB

```
import lldb
```



The screenshot shows the Xcode IDE with a Python script named CustomSummaries.py open in the editor. The script defines a function circle_summary and an __lldb_init_module function. The circle_summary function calculates the area of a circle based on its width and height. The __lldb_init_module function registers the circle_summary function with LLDB. The Xcode interface includes a toolbar at the top with Run, Stop, Scheme, and Breakpoints buttons, and a status bar at the bottom with a search and refresh icon.

```
def circle_summary(valueObject, dictionary):  
    # grab the bounds of the SKTCircle.  
    bounds = valueObject.GetChildMemberWithName('_bounds')  
    size = bounds.GetChildMemberWithName('size')  
  
    # grab the width and height form the circle's bounding NSRect.  
    error = lldb.SBError()  
    width = size.GetChildMemberWithName('width').GetData().GetDouble(error, 0)  
    height = size.GetChildMemberWithName('height').GetData().GetDouble(error, 0)  
  
    # calculate the two radius' of the ellipse.  
    radiusOne = width / 2.0  
    radiusTwo = height / 2.0  
    area = math.pi * radiusOne * radiusTwo  
  
    # capture the major and minor radius.  
    majorRadius = radiusOne if radiusOne > radiusTwo else radiusTwo  
    minorRadius = radiusOne if radiusOne < radiusTwo else radiusTwo  
  
    # generate the strings we'll use in the summary.  
    majorRadiusAsString = str("{0:.1f}".format(majorRadius))  
    minorRadiusAsString = str("{0:.1f}".format(minorRadius))  
    areaAsString = str("{0:.1f}".format(area))  
  
    return 'Major Radius = ' + majorRadiusAsString + ', Minor Radius = ' +  
        minorRadiusAsString + ', Area = ' + areaAsString  
  
# this is a special funtion, which if exists, will be called by LLDB when this script is  
imported.  
def __lldb_init_module(debugger, dict):  
  
    # automatically add our summaries when the script is imported.  
    debugger.HandleCommand('type summary add SKTCircle -F CustomSummaries.circle_summary')
```


Creating Custom Summaries

LLDB

```
import lldb
```

```
def circle_summary(valueObject, dictionary):
```

```
def circle_summary(valueObject, dictionary):
    # grab the width and height from the circle's bounding rectangle.
    error = lldb.SBError()
    width = size.GetChildMemberWithName('width').GetData().GetDouble(error, 0)
    height = size.GetChildMemberWithName('height').GetData().GetDouble(error, 0)

    # calculate the two radius' of the ellipse.
    radiusOne = width / 2.0
    radiusTwo = height / 2.0
    area = math.pi * radiusOne * radiusTwo

    # capture the major and minor radius.
    majorRadius = radiusOne if radiusOne > radiusTwo else radiusTwo
    minorRadius = radiusOne if radiusOne < radiusTwo else radiusTwo

    # generate the strings we'll use in the summary.
    majorRadiusAsString = str("{0:.1f}".format(majorRadius))
    minorRadiusAsString = str("{0:.1f}".format(minorRadius))
    areaAsString = str("{0:.1f}".format(area))

    return 'Major Radius = ' + majorRadiusAsString + ', Minor Radius = ' +
        minorRadiusAsString + ', Area = ' + areaAsString

# this is a special function, which if exists, will be called by LLDB when this script is
imported.
def __lldb_init_module(debugger, dict):

    # automatically add our summaries when the script is imported.
    debugger.HandleCommand('type summary add SKTCircle -F CustomSummaries.circle_summary')
```

Creating Custom Summaries

LLDB

```
import lldb
```

```
def circle_summary(valueObject, dictionary):
```

SBValue

```
Sketch.xcodeproj - CustomSummaries.py
Finished running Sketch.app : Sketch
Project 1
Editor View Organizer

def circle_summary(valueObject, dictionary):
    # grab the width and height from the circle's bounding rectangle.
    error = lldb.SBError()
    width = size.GetChildMemberWithName('width').GetData().GetDouble(error, 0)
    height = size.GetChildMemberWithName('height').GetData().GetDouble(error, 0)

    # calculate the two radius' of the ellipse.
    radiusOne = width / 2.0
    radiusTwo = height / 2.0
    area = math.pi * radiusOne * radiusTwo

    # capture the major and minor radius.
    majorRadius = radiusOne if radiusOne > radiusTwo else radiusTwo
    minorRadius = radiusOne if radiusOne < radiusTwo else radiusTwo

    # generate the strings we'll use in the summary.
    majorRadiusAsString = str("{0:.1f}".format(majorRadius))
    minorRadiusAsString = str("{0:.1f}".format(minorRadius))
    areaAsString = str("{0:.1f}".format(area))

    return 'Major Radius = ' + majorRadiusAsString + ', Minor Radius = ' +
        minorRadiusAsString + ', Area = ' + areaAsString

# this is a special funtion, which if exists, will be called by LLDB when this script is
imported.
def __lldb_init_module(debugger, dict):

    # automatically add our summaries when the script is imported.
    debugger.HandleCommand('type summary add SKTCircle -F CustomSummaries.circle_summary')
```

Creating Custom Summaries

LLDB

```
import lldb
```

```
def circle_summary(valueObject, dictionary):
```

Don't use this
parameter

```
def circle_summary(valueObject, dictionary):
    # grab the width and height from the circle's bounding rectangle
    error = lldb.SBError()
    width = size.GetChildMemberWithName('width').GetData().GetDouble(error, 0)
    height = size.GetChildMemberWithName('height').GetData().GetDouble(error, 0)

    # calculate the two radius' of the ellipse.
    radiusOne = width / 2.0
    radiusTwo = height / 2.0
    area = math.pi * radiusOne * radiusTwo

    # capture the major and minor radii
    majorRadius = radiusOne if radiusOne > radiusTwo else radiusTwo
    minorRadius = radiusOne if radiusOne < radiusTwo else radiusTwo

    # generate the strings we'll use in the summary.
    majorRadiusAsString = str("{0:.1f}".format(majorRadius))
    minorRadiusAsString = str("{0:.1f}".format(minorRadius))
    areaAsString = str("{0:.1f}".format(area))

    return 'Major Radius = ' + majorRadiusAsString + ', Minor Radius = ' +
        minorRadiusAsString + ', Area = ' + areaAsString

# this is a special function, which if exists, will be called by LLDB when this script is
imported.
def __lldb_init_module(debugger, dict):

    # automatically add our summaries when the script is imported.
    debugger.HandleCommand('type summary add SKTCircle -F CustomSummaries.circle_summary')
```


Creating Custom Summaries

LLDB

```
import lldb
```

```
def circle_summary(valueObject, dictionary):
```

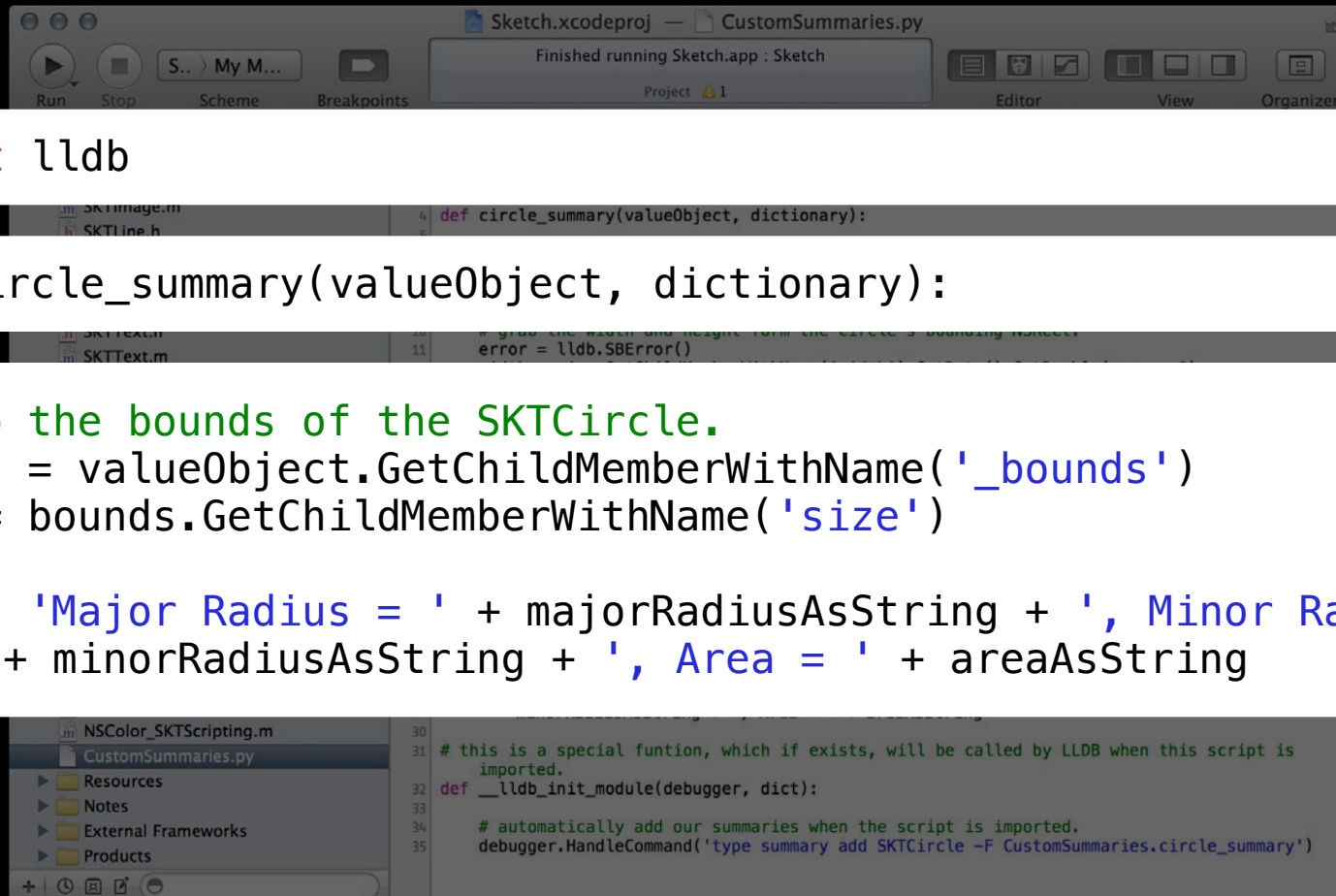
```
# grab the bounds of the SKTCircle.
```

```
bounds = valueObject.GetChildMemberWithName('_bounds')
```

```
size = bounds.GetChildMemberWithName('size')
```

```
...
```

```
return 'Major Radius = ' + majorRadiusAsString + ', Minor Radius =  
    ' + minorRadiusAsString + ', Area = ' + areaAsString
```



Demo

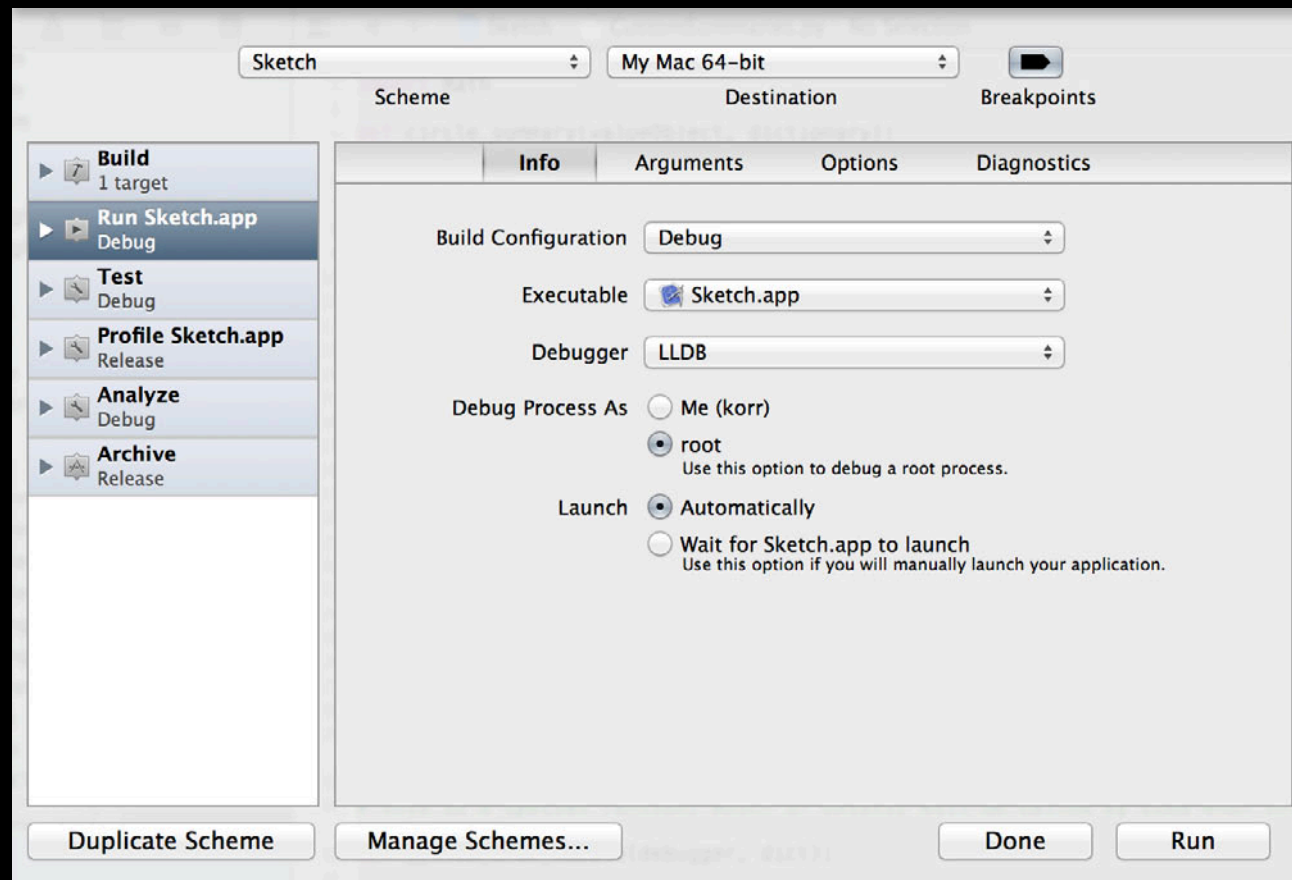
Variables view and custom LLDB summaries

Troy Koelling

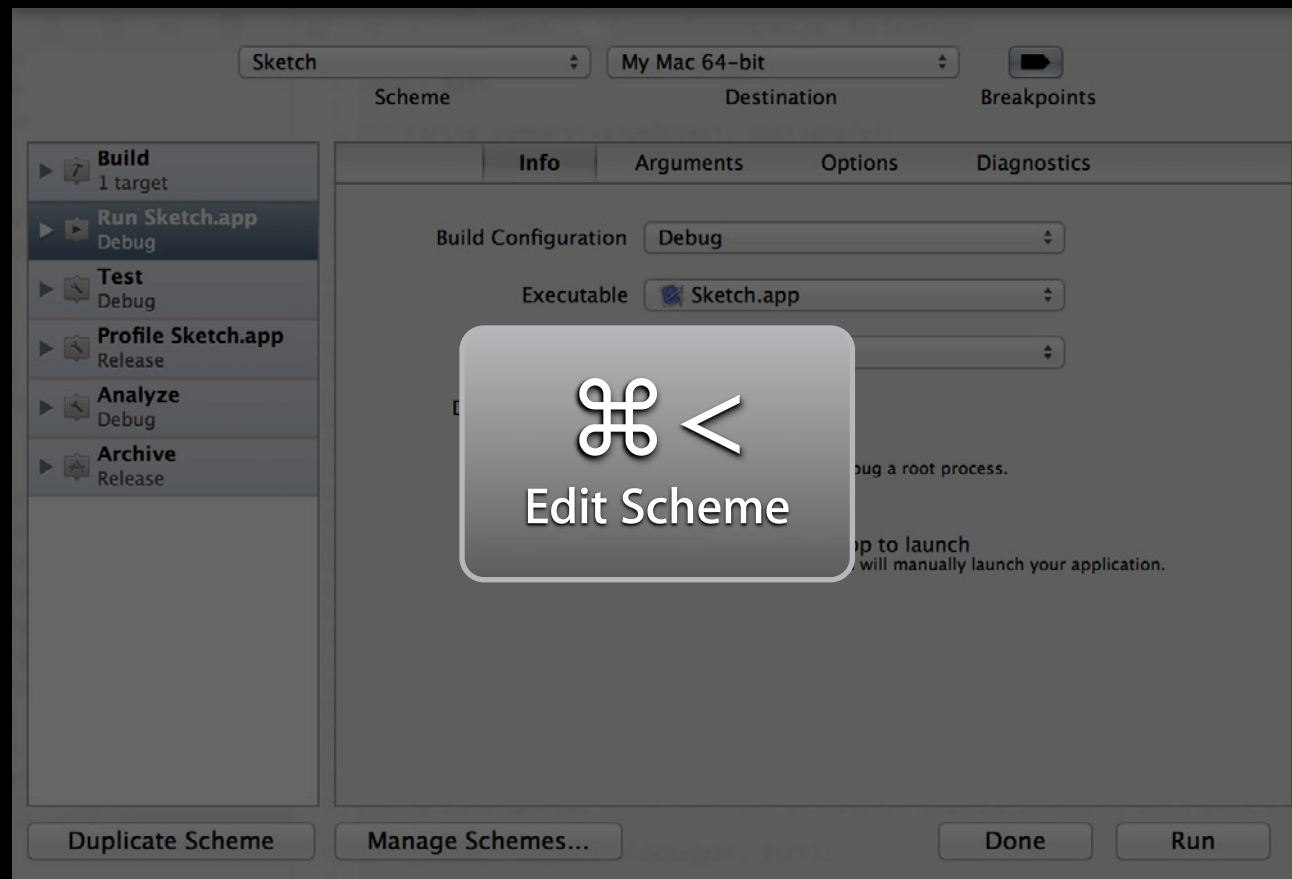
Xcode Debugger UI Engineer

Advanced Debugging

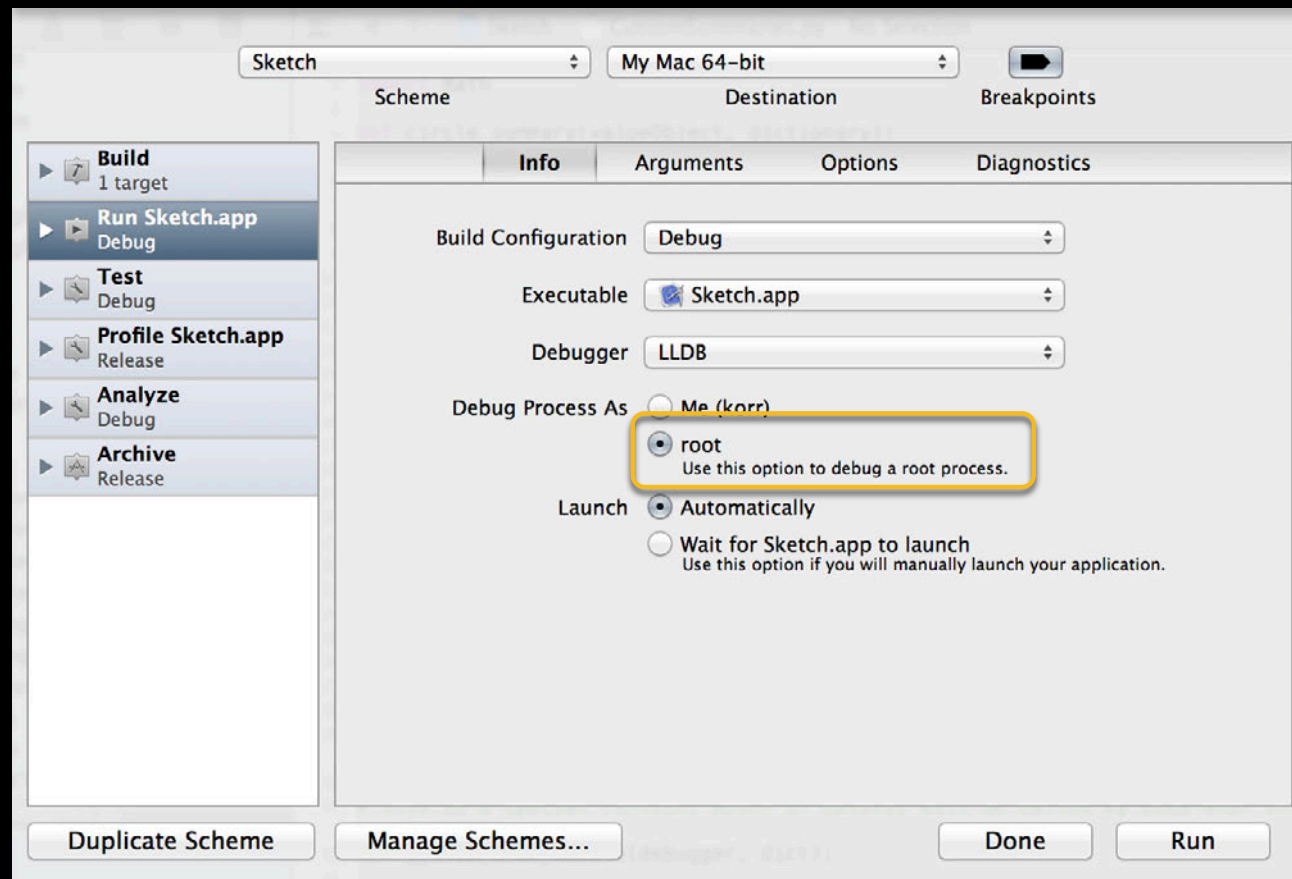
Debugging as Root



Debugging as Root



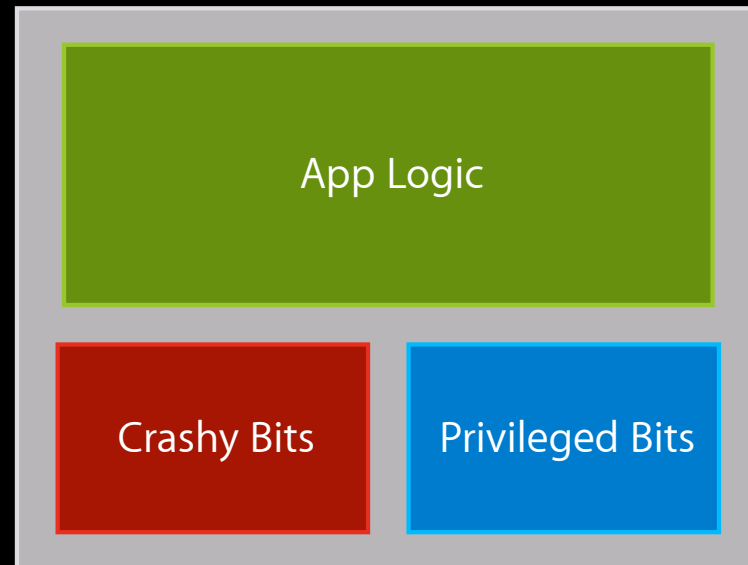
Debugging as Root



What Is an XPC Service



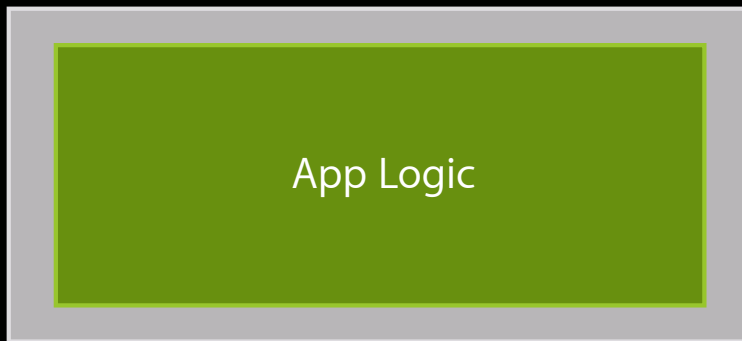
Your App Process



What Is an XPC Service



Your App Process



XPC Services
(separate processes)

Crashy Bits

Privileged Bits

Demo

Advanced debugging

Han Ming Ong

Xcode Debugger UI Engineer

More Information

Michael Jurewitz

Developer Tools Evangelist

jury@apple.com

Documentation

LLDB Custom Summaries

<http://lldb.lvm.org/varformats.html>

Documentation

LLDB Scripting

<http://lldb.lvm.org/scripting.html>

Apple Developer Forums

<http://devforums.apple.com>

Related Sessions

OpenGL ES Tools and Techniques

Pacific Heights
Wednesday 3:15PM

Learning Instruments

Presidio
Wednesday 4:30PM

Cocoa Interprocess Communication with XPC

Russian Hill
Thursday 4:30PM

Debugging with LLDB

Presidio
Friday 10:15AM

Labs

Xcode Lab

Developer Tools Lab B
Ongoing

LLDB Lab

Developer Tools Lab C
Friday 11:30AM



Unaligned Lines
& Flashes

Provision structural
member details, their
style, & background
colorings

working
table

Color palette

Transparenc
effects

working table
Transparenc
effects
color tool

EW

ARCHITECT: XCODES

#1

PROJECT: APPLICATION_APP

 **WWDC2012**