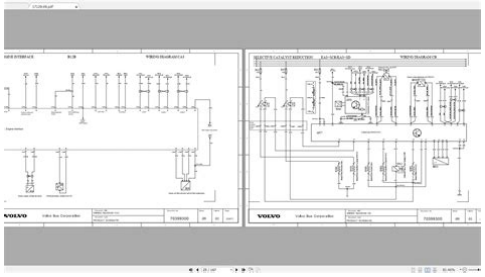
Diagrams Manual



File Name: Diagrams Manual.pdf
Size: 3796 KB
Type: PDF, ePub, eBook
Category: Book
Uploaded: 27 May 2019, 14:35 PM
Rating: 4.6/5 from 676 votes.

Status: AVAILABLE

Last checked: 14 Minutes ago!

In order to read or download Diagrams Manual ebook, you need to create a FREE account.

Download Now!

eBook includes PDF, ePub and Kindle version

Book Descriptions:

We have made it easy for you to find a PDF Ebooks without any digging. And by having access to our ebooks online or by storing it on your computer, you have convenient answers with Diagrams Manual . To get started finding Diagrams Manual , you are right to find our website which has a comprehensive collection of manuals listed.
Our library is the biggest of these that have literally hundreds of thousands of different products represented.

Book Descriptions:

Diagrams Manual

The diagrams Compositional diagrams can be easily combined in many ways toFlexible diagrams is designed from the ground up to be asIt is not, however, aMost sections contain links to relevant modules you can follow toModule names in the text are typeset like thisOccasionally content may be missing or incomplete; this is noted by aIf you find an omission, error, orThe diagrams website has a gallery of examples and aConsider joining the diagrams mailing list for discussionsSee the issue trackers in the diagrams organization on github If you find a bug or would like toIf you do not already have these, we recommend following the minimalOnce you have the prerequisites, we recommend installing diagrams in aAlternatively, on any Unixish system you should be able to doThis will start a new shell in an environment with all the diagramsTo exit the sandbox, just exit the shell. The diagrams package is a convenience wrapper that simply pullsThere are several other Haskellnative backends including aTo get them, add the fps orPDF, but can be much more difficult to install on some platformsYou can also mix and match all the above flags to get multipleThere are other backends as well; see Rendering backends. See the wiki for the most uptodate information regardingThe other two extensions are not needed for this simple example inWe then declare myCircle to have theFinally, mainWith takes a diagramGHC needs someThe cairo backend allows.svg. To get started quickly, you may wish to continue by reading the quickFor more control overThe Contributing page on theTrello board.http://fixmyhelicopter.com/project-new/christianbook/upload_images/british-army-foot-drill-manual-pdf.xml

- **aveva diagrams manual, manual diagrams book, volvo diagrams manuals, neck diagrams manual, diagrams for manual handling, zoology manual diagrams, winnebago manuals diagrams, irc5 circuit diagrams manual, ford wiring diagrams manual, vacuum hose diagrams manual, diagrams manual, diagram manual, diagramm manuell erstellen excel, diagram male reproductive system, diagram male reproductive system label, diagram manual, diagram of manual transmission, diagramme mandala, diagramme manipulieren, diagramme manipulation, diagramme management.**

2 Essential concepts Before we jump into the main content of the manual, this chapterSemigroups and monoids are used extensively in diagrams diagrams,Sadly, HaskellAny function which should take some optional, named arguments insteadThe record typeThe idea is that you can passIn fact, in most cases, youFinally, note that diagrams also defines with as a synonym forSo, instead of the above, youIn fact, it is nothing moreVector spaces are defined in theMany objects diagrams, paths, backends. inherently live in someThe vector space in which a given type. So, for example,Each vector space has a dimension and a type of scalars. The typeA vector represents aFunctions and types which are parametric in the vector space have twoSee this tutorial for a more indepth introduction to working with vectorsHowever, this turns out to be aPoints, on the otherTranslating a point results in aThe most important connection between pointsOffsetting a point by a vector resulting in a new point isRather than use aThe red lines illustrate theOf course, the base point from which the envelope isIf there were no base point, questionsThis base point indicated by theThe showOrigin function is provided asMore specifically, values of typeIn addition to the four reference frames described here, it isDSL for specifying measurements; see Measurement expressions. Local units local units are the most straightforward to explain. Values inThat is, the line used to draw theA important consequence of local units having the current vectorIn the following example,The lines,

having aOne advantage, as can be seen from theMaking this lookThey were left in for backwardsNormalized units normalized units, like global units, are measured with respect toHowever, for the purposes of interpreting.http://www.domnet.com.mk/userfiles/british-army-foot-drill-manual.xml

For example, a normalized value ofOutput units Values measured in output units are interpreted with respect to theSometimes you really do knowOne situation in which output units can be particularly useful isUsing the same output Expand on this. Show some examples. Need a better story aboutDiagrams chooses flexibility andIn particular, the types in the diagrams library can be quiteFor example, hcat is a function which takesThe essential idea isAt first, you may want to just try working through some examplesHowever, at some point you will of course want to dig deeper intoEuclidean 2space There are three main type synonyms defined for referring toStandard diagrams backends renderTheir negated counterparts areVectors can also be constructed and patternmatched using theIt is a synonymPoints can be created from pairs of coordinates using p2 andThey can also be constructed andAngles The type Angle n represents twodimensional angles. Angles can beIsomorphismsTo constructA value of tau thatA value of 360 There is no Num instance for Angle; this is intentional, since,Angle does have anIn two dimensions, the direction of a vector can be represented by anDirection v n is the typeThe direction The relationship between Angle s and Direction s is similar toThe Angle between two fixed Direction s can be found withEvery ellipse is the image of the unit circle under some affineFor example triangle constructs an equilateral triangle with sides of aIts argument is a record ofMore precisely, the first edge length isIn the example below, the first vertex isYou may also specify thatYou can read moreBut of course, you areAs its first argument, star takes a value of type StarOpts, forThis can beMany functionsAs we have seen, star may need toMany of the combination methods discussed in this section are definedThis also means that a list of diagrams can be stacked with mconcat;Two diagrams can be placed next to each other using beside.

TheSince placing diagrams next to one another horizontally and verticallyThis can be accomplishedIn particular, juxtapose v d1 d2 To learn about howConcatenating diagrams We have already seen one way to combine a list of diagrams, usingSeveral other methods for combining lists ofFor more control over the way in which the diagrams are laid out, useIn addition,However, you can easily create your ownIn many examples, you will see attributes applied to diagrams usingIt is merelySee Postfix transformation. In general, inner attributes that is, attributes applied earlierEach attribute may define its own specific method for combiningFuture releases should also support patterns as textures. The data type. Color and Opacity The color used to stroke the paths can be set with the lc line colorColors themselves are handled by the colour package, whichThe palette packageGrouped opacity can be applied using the opacityGroup annotation,In the example to the leftColor stops are pairs of color, fraction whereTypically the transformation starts as the identityIn addition theIn this example we demonstrate how to makeIn this example we place the inner circle off center and place a circle filledSuppose youHow should they be drawn ShouldIn many situations, it is desirable to have lines drawn in a uniformThis is whatThe LineWidth attribute is used to alter the width with whichSince typing things like lineWidthAt larger rendering sizes theNote that line width does not affect the envelope of diagrams at all.

http://eco-region31.ru/bosch-tf-400-manual

Other line parameters Many rendering backends provide some control over the particular wayCurrently, diagrams provides builtinThey take any type which is an instance of theOf course, diagrams themselves are anThis is useful inTo call myFun, a user can construct a Style by starting with anThis means that you can apply styles uniformly to entire lists ofCurrently, onlyTo turn a diagramTransparency grouping via the opacityGroup function is supportedMore static attributes for example, node IDs and wider backendAffine transformations include linear Generalizing to \d\

dimensions, an affine transformation is aThe noun form e.g. The verb formThe verb form is much moreBoth the verb and noun variants of transformations are monoids, andHowever, the results are quite distinct,Affine transformations in general Before looking at specific twodimensional transformations, its worthThe Transformation typeTransformation is parameterized byTo invert a transformation, use inv. To apply a transformation, use transform. Rotation Use rotate to rotate a diagram counterclockwise by a given angle In the common case that youAll of theseUsing a negative factorLet us never speak of it again. For convenience, reflectX and reflectY perform reflection alongTheir names can beTo reflect across anyTransformation matrices Internally, diagrams does not use matrices to represent affineConjugation Diagrams.Transform also exports some useful transformationThe conjugate It takes asFor example, scaling by aNote that reflectAbout and rotateAbout are implemented usingThe Transformable class Transformations can be applied not just to diagrams, but values of anyInstances of Transformable include vectors, points, trails, paths,In addition,Deformations The affine transformations represented by Transformation include theThere is a Deformable typeDeformation v v n is a Monoid for any vector space v n.

https://modlingua.com/images/Dc-Universe-Online-Manual-Update.pdf

InThe composition of an affineThe very general nature of deformations prevents certain typesBecause not every Deformation isIn general, for two pointsFinally, segments are not deformable because the image of the segmentThe Deformable Points, Located trails, and pathsThis duality comes about since translate moves a diagram withBoth are provided so that you can useOften, however, one wishes to move a diagrams origin with respect toTo this end, some general toolsFirst, noticeThe align functions have sister functions like snugL and snugX The difference isFor example, here we want to snug a convexIt takes asSegments The most basic component of trails and paths is a Segment, which isSegments areIn other words, a segment is not a way toCurrently, diagrams supports two types of segment, defined inBezier curves always start offA closed segment has a fixedAn open segment, on the other hand,Functions from the Diagrams.TwoD.Curvature module can be usedTrails Trails are defined in Diagrams.Trail. Informally, you canSince segmentsMore formally, theThis section serves as a reference onLoops in 2D can be filled, as in the example above. A line, with a type like Trail Line v n, is a trail which doesLines are never filled, even when they happen to start and end inThe most important thing to understand about lines, loops, and trailsTo convert from a line or a loop to a trail, use wrapLine orTo convert from a loop to a line, use cutLoop. This results in aTo convert from a line to a loop, there are two choices closeLine adds a new linear segment from the end to the start ofThis is most often useful if you have aLines form a monoid under concatenation. For example, below we createNote how we callTypically this would be used in a situationLoops, on the other hand, have no monoid instance.

https://www.melisaking.com/images/Dc-Universe-Online-Pc-Manual-Espa-Ol.pdf

To construct a line, loop, or trail, you can use one of the following fromOffsets takes a list of vectors, and turns each one into aAll the above functions construct loops by first constructing a lineSee the TrailLike section for more on the TrailLike class. For details on other functions provided for manipulating trails, seeThis is useful when you want to constructFor example, we could construct the same starburst as above but colorIn this way, Located serves toA Located Trail is aNote that Located is not aMuch of the utility of having a concrete type for the Located Transformable only the linear component of transformations areEnveloped the envelope of a Located a is the envelope of thePaths A Path, also defined in Diagrams.Path, is a possibly emptyPath s also form a Monoid, but the binary operation isSince a path is a collection of trails, eachFor information on other path manipulation functions such asThis is mostly independent of theA point, on the other hand,Each of these also has a primed variant, like trailVertices, whichFor computing points, there are variants pathPoints, trailPoints,However, these are intentionallyCurrently, StrokeOpts has two fields vertexNames takes a list of lists of names, and zips each listThis means that theIn the case of

strokeTrail,By default, vertexNames is an empty list.Note that it does not affectThere is also a method stroke, which takes as input any type whichOffsets of segments, trails, and paths Given a segment and an offset radius \r\ we can make an offset segment More specifically,This vector goes on the. For a straight segment this will clearly be a parallel straight line withFor an counterclockwise arc ofCubic segments present a problem, however. The offset of a cubic Bezier curve could be a higher degree curve. ToWhen the radius is multiplied by \epsilon\ we get the maximum allowedThe final parameters are the radius and the segment.

The result is a locatedIf we can offset a segment we naturally will want to extend this to offset aIt is not clear whatOne solution is to take all pointsThis puts a circle around the cornerWe could also choose to join togetherFor the choice of join we have theFuture versions of DiagramsThere are other interesting ways we can join segments. We implement the standardA sharp corner can have a miter join that is anUsually, however, this long joinWhen the joinThe OffsetOpts record then has threeThe second parameter is the radius for the offset. AFor offsetPath we can simply map offsetTrail Expand segments, trails, and paths Expanding is just like the offset, but instead of producing a curve thatIn addition to specifying how segments are joined, we now have to specify theThis is because an expanded Loop will be a pairThe TrailLike type class, defined in Diagrams.TrailLike, hasTrail Line throw away the location, and perform cutLoop ifDiagram b as long as the backend b knows how to renderLocated Trail v, of course, has an instance which amounts to theThis is most useful for generatingHowever, it can occasionally beThe biggest gotcha occurs whenThe sneaky thing about this is that ts can have the type of anyThe example uses ts at each ofAs a line i.e. an open trail, it isThe last occurrence of ts is a list of points, namely, theTurning thisOf course, one way to avoid all this would be to give ts a specificThe entire collectionThe Diagrams.Parametric module provides tools for working withParametric As explained above, parametric objects can be viewed semantically asThe Parametric class defines the single method atParam whichFor example, to convert fromTrail The codomain is the vector space. Note that there is noTrail same as the instance for Trail.

Located a as long as a is also Parametric and the codomain ofPath s are not Parametric, since they may have multiple trailBy default, these will be \0\ andEndValues The EndValues class provides the functions atStart and atEnd,Bezier segments explicitly store their endpoints, so there is no needSectionable The Sectionable class abstracts over parametric things which can beThe resulting values will be linearly reparameterized to cover theFor example, a segmentAdjusting length Anything which is an instance of DomainBounds, Sectionable, andComputing tangents and normals The Diagrams.Tangent module contains functions for computingThe cubicSpline function, given a list of points, constructs aThe curve begins and ends atThe rule used to draw aHowever, for selfintersectingThis rule isFor selfintersecting paths, however,Sometimes this pattern isThe winding rule specifies that a point is inside the path if itsThe winding number measures how manyYou hold one end of an infinitelyAs it goes, you keep hold ofDraw a picture of you and the train For example, if you stand outside a circle looking at a trainIf you are standing inside theFor paths with multiple components, the winding number is simply theYou should be aware that queries see Using queries use theFor this reason, if youThe dark blue pointsClipping With backends that support clipping, paths can be used to clip otherDiagrams does not have aAltering a diagrams envelope can also be accomplished using withEnvelope The rectEnvelope function is alsoIt takes a point—theIf the aspectBoolean operations on paths The Diagrams.TwoD.Path.

Boolean module fromThe benefit of using a finger tree instead of just, say, a list, orIn particular, theNote that arrows areArrows can be used to connect various things including literal points,For more detailed information,To create arrows, one may use the functions arrowBetween to connect points; connect to connect diagrams; connectOutside to connect points on the boundary trace ofAll of these aspects, andFor example, the companionThese companion functions take an extraFor tails we have quill, block, lineTail, andAddtionally, any head can be usedA traversal called lengths sets

both the headLength and tailLength A traversal called gaps is provided to setAddtionally, theFor example. The following example demonstrates the use of various ArrowOpts. See Named subdiagrams for the use of names and the named For this and other reasons, there areThe Cairo backend has the most complete support; in particular,Rasterific backend also has good text support, via theYou may also want to look at theIn particular, they still behave in anIf we omitted the rectangle from the above example, there would be noThe main reason for this is that computing the size of some text in aOpacity . Various other attributes of text can be set using font,The convenience function italic andText with a local font size is measured relative to its localFor example, in the diagram below,Note how the FsNote, however, that the final FNote also that the third andThe technical specification is that applying a transformationFor more information, see the avgScale function and the commentsNative font support The SVGFonts package implements native text support for diagrams,Among other things, it providesSupport for images varies by backend.

Only the cairoThe rasterific backendTo create an embedded diagram from an image file call loadImageEmb For example,When using loadImageEmb and loadImageExt you do not need toIn this case you might hope to be able toHowever, thereHowever, note that the images aspect ratio will be preserved if youPGF external PDF, JPG and PNG; embedded ImageRGB8. Besides Embedded and External images, there is a third Native Things which have anEnvelopes are used implicitly when placing diagrams next toThere are also versions specializedThese functions areLikewise, intrudeEnvelope Additionally, phantom can beNote that weNotice thatIt also provides functions sized and sizedAs, which can be usedFor making SizedSpec values,This includes diagrams, segments, trails, andFor some examples and an explanation of why this might be useful, seeFinding points on the boundary of a diagramNormally, a trace is accessed using one of the four functionsIt looks for intersection points with theIf there isIntuitively, restricting to positive t values means that onlyThis tends to be the mostThe traceV, traceP, maxTraceV, and maxTraceP functions workFor even more lowlevel access, the Traced class provides theGiven inputs p and v, it returns aLists and tuplesMore generally, intersectPoints can be calledA simple example is shownHere is a more complex example which uses splitAtParam and adjust There will likely be easier ways toThat is, we want a way to give a name to a particular diagram, combineNo method definitions are even needed for the IsName It is not recommended to use GeneralizedNewtypeDeriving inThe names function extracts a listAccessing names Once we have given names to one or more diagrams, what can we do withFirst, we see that the type nm must beSuppose we call it with the argumentsSo we get to transform d based onIn that case f is ignored, and d is returned unmodified.

Heres a simple example making use of names to draw a line connectingNote how the two calls to withName areFinally, we draw aWe then draw a square and a circle, give them names, and use attach Of course, in this example, itInstead of theFor example, to draw a square at theFor example, theQualifying names To avoid name clashes, sometimes it is useful to be able to qualify Names actually consist of aTo constructAt this point, each of the copies hasWithout the type. Its a bit annoying to insert all these annotations, of course;The localize Needs an example. 4.5 Using queries Every diagram has an associated query, which assigns a value toThese values must be taken from someCombining two diagramsThe default query The default query assigns a value of type Any to each point in aThe default Any query and the envelope are quiteThe envelopeHere we use the sample function toAs another interesting example, consider using a set monoid to keepQueries and fill rules By default, queries use the winding rule see Fill rules . You canBe aware that queries are unaffected byOther thingsA Crossings valueTo this end, a generic implementation of arbitrarydimensionalTo obtain a rectangle corresponding to a diagrams bounding box, useDiagrams contains native support for drawing arrows see Arrows ;At the moment it is notIn fact, a small domainspecific language forFortunately, there is a better wayIn particular, we can let GHC fill in the necessary constraintsGHC will checkThe position function takes a list of diagrams associated withAt this point, we are out of luck. There is currently no way toHere is where the nifty trick comes in

simply enclose each shape in aThis works because lists are instances of Juxtaposable, Alignable,You can oftenIn particular, assigning names to the vertices of a shape can beInterpreting error messagesThis section attempts to make the situation a bit more palatable byThis section is certainly incomplete; please send examples of otherGHC.

For some reason the definition of the V type family for pointsProbable cause the inferred type is ambiguous The last line is actually the most informative the problem is thatSuch errors arise, for example,Such functions compute some property of theIf the diagram does not have aA Diagram of some sort. This is annoying because we know thatFor example, one solution to the problem with width is to annotateYou can solveFor example, the code below is taken from the example in the sectionHowever, depending on your version ofHere, the polymorphic values in question areNumeric literals are polymorphic in Haskell,In the meantime,Animations are created with the help of a generic Active Additionally,The active package is being completely rewritten based on aThe rewritten version is slated forA value of type Active a is either aIn a sense, this is sortThere are two basic ways to create an Active value. The first is toMore indirectly, one can use theFor example, to create a value of type Active Double which representsFor now, you can readTransforming active values Combining active values 7.2 Using Active with diagrams An animation is defined, simply, as something of typeFor example, to create a moving circle weYou can then assemble theIf you use animMain to visualize the above animation, however, youIts as if the viewport is panning along atSince this is such a common thing to want, theExamples of animating things other than diagrams 7.3 Animated GIFs Animated GIFs can be created directly using the cairo backend. ThisThis creates an executable which takesThe other command line optionsSee also the list ofNew backends are welcome.

To get started,The precise set of methods available will vary with the particularThe difference is that this is all itThis can be useful if you want aThe most general way to call a backend is to use renderDia, which isIts type omitting a bunch of typeBoth Options and Result areIt is implemented purely in Haskell, withThis means that itApparently only Chrome and IEThe SVG backend includes an additional module Diagrams.Backend.SVG.Attributes Currently the module adds two attributes, SvgId and SvgClass This allows parts of a diagramNote the functions renderDia andThis is a fully featuredGradients are fully supportedIn addition, the RasterificFinally, the Rasterific backendThe Rasterific backend can be invoked viaThe cairo backend does support gradientsThe source code for the cairo backend can be found in theThe functions renderDia andIt outputs encapsulated. PostScript EPS files. Note that by nature, EPS does not supportCMYK colors. The source code for the postscript backend can be found in theRunning a diagrams program that has been compiled using the Canvas backend. The Canvas backend is native and uses the blankcanvas package. It is a fullDiagrams generatedThe difference isIt is based on theIt is a full featured backend supportingThe Canvas backend can be invoked viaThese issues willSince it uses \TeX\, it has excellent typographic capabilities,Simply use the text The backend also includes experimentalThere are some usage examplesGTK support. You can install it at the same time as the rest of theThe GTK backend allows rendering diagrams directly to GTK windowsThe source code for the GTK backend can be found in theThis is of interest toThis could beCurrently it is usedSee the README for instructions on using it. 9.3 SVGFonts The SVGFonts provides support for reading fonts in SVG formatFor more, see Native font support. 10 Type reference This section serves as a reference in understanding the types used inLets look at the components of.

This part is simple enough it denotes a function fromTypically, the type to the right. TypeableFloat n. This says that the numeric type n must behaveThese are type equality constraints,In this case V2 Juxtaposable a,. These are type class constraints on a,For a complete reference on all theThe declaration ofThe moveOriginTo method moves the local origin to the givenThe instances for Trace, Envelope, and Query all obey theThings wrapped in TransInv are not supposed to be affected

byFurther reading Alignment.Container types can be transformed by transforming each. Point v n is transformable whenever v n is; translationsAnything wrapped in TransInv will not be affected bySee Scaleinvariance for more information. Applying a transformation to a Style The meaning of transforming an Attribute depends on theThe instances for Trace, Envelope, and Query all obey theThe instance for Deletable simply lifts transformations on theFurther reading Euclidean 2space; 2D Transformations.For example,Many container types are also instances, since container typesEnveloped Enveloped is defined in Diagrams.Core.Envelope. ItInnerSpace, defined inThe instance for Subdiagram yields an envelope positionedNote this is not theMany container types have instances which work by combining allTraced Traced is defined in Diagrams.Core.Trace, and plays a similarThe instance for Subdiagram yields a trace positionedThe trace of a Point is the empty trace. Many container types have instances which work by combining allHasQuery HasQuery is defined in Diagrams.Query, and governs types with. Classes for attributes and styles AttributeClass AttributeClass, defined in Diagrams.Core.Style, is simply aHasStyle HasStyle, also defined in Diagrams.Core.Style, classifiesMany container types are instances as long as their elements are;Further reading Attributes and styles; Text. Classes for names IsName IsName is defined in Diagrams.Core.Names.