**Properties：**

1. **DrawFigures**：this object includes functions to draw custom graphics, such as createText, createLine, etc; Use can call these functions to draw extra graphic element to meet special needs.
2. **GraphType**：this property record current graphic type, such as ‘Pie’,’Radar’,etc.
3. **ID**：this property saves an ID string to identify DChart of current usage.
4. **Language**：this property saves language set.
5. **calculatedBasic**：this object saves information of basic sets, include size, margin of canvas.
6. **canvas**：this object is the DOM object that is created inside parent div DOM.
7. **coordinates**：this object collects coordinate information that every graphic element was drawed, such as size, location,etc.
8. **ctx**：object get from canvas.getContext('2d')
9. **innerData**：this object saves data that user passed(if user use pure-array data, then innerData saves normal object data that DChart transferred from pure-array data).
10. **innerOptions**：this object saves options that user passed.
11. **originalDefaultOptions**：just like the name, this object saves original default options. In every js that belongs each graphic type, default options are extended by ‘this.innerOptions = DChart.Methods.Extend(this.originalDefaultOptions, {})’.
12. **randoms**：this object saves random strings that is useful in process of drawing graphics.
13. **shapes**：this array saves temp shapes in process of drawing graphic. It is used to redraw part or whole graphic, get coordinate information, response mouse events, etc, important useages.
14. **tempData**：this object saves temp but important variables that are used as configuration. These variables may be the special set that [type.js] need to pass to [core.js], or may be the computed result that saved as ‘cache’ data.

**Events**：

1. **onStart**：Built-in event, executed when DChart ‘start to work’.
2. **onBeforeAnimation**：Built-in event, executed when DChart begin animation.
3. **onAnimation**：Built-in event, executed when animation is on. A parameter is pass to the function that show percent of animation of completion.
4. **onFinish**：Built-in event, executed when DChart finish current drawing task.

**Functions**：

1. **ClearBackGround**：Clear all elements in canvas, include canvas sketchpad, spans to show tips, saved shapes, etc.
2. **Draw：**Key function, to start drawing graphics. This function has two parameters: data and options. User can pass one them, or neither of them only if make sure that: when data is not passed, function SetData must be called before; when options is not passed, function SetOptions must be called before.
3. **GetCoordinate**：Get coordinate information of graphic elements. Parameter location is the “path string” to locate the coordinate of what user need to get. “.” Is used to express child items, such as “canvas.height” stands for height of canvas. User can use “.[index]” to get coordinate info in array format, for example, “pie.outerlabels.3.left” stands for left location of forth outer label of pie’s semicircles.
4. **Initial**：Re-initialize canvas element. Works include delete old canvas element(if exists), delete old styles in DCharts, add new styles, compute basic coordinate information of canvas, add new canvas element, initial configs, clear canvas, etc. In summary, this function is to initial a new canvas to draw graphics.
5. **SavePic**：Export picture of canvas(Some browsers do not support). Parameters are filename and picture type, such as png,bmp,gif.
6. **SetBasicOptions**：Change basic config for DChart. This function can change size of canvas, therefore when this functions is called, Initial will be called inside.
7. **SetData**：Load data. This function specially for loading data, Independent of Draw function, the effect is consistent with loading data in Draw function.
8. **SetDefaultOptions**：Set back to default options, include common options and specific options.
9. **SetOptions**：Load options, Just like SetData, independent of Draw function.
10. **SetSkin**：Set skin for DChart. If user need to guarantee that options through normal options have high priority than through skin, he must call SetSkin before SetOptions or Draw.
11. **\_calculateOutersValid**：Collect information if each external element shall be drawed, such as title, legend, axises.
12. **\_changeTip**：Change the location of single tip. If the location of tip is Improper(such as covering graphics, beyond the boundary), then the location needs to be adjusted.
13. **\_checkOptions**：Check if loaded options meet the requirements. For example, user cannot pass a string to one option that needs a number.
14. **\_clearDrawable**：Clear drawable part of canvas. This function only clears canvas sketchpad(doesn’t delete tips, shapes, etc) . And the target sketchpad is just used for animation, the remaining rectangular area after removing the peripheral region include title, legend, axises, etc.
15. **\_clearTips：**Clear all Dom objects for tips.
16. **\_computeAxis：**Compute coordinate information of axises, to draw axises, scales, and other elements of graphic.
17. **\_computeBasic：**Compute basic config of canvas. If SetBasicOptions is called, this function will be called to recalculated.
18. **\_computeLegend：**Compute coordinate information of legend.
19. **\_computeTitle：**Compute coordinate information of title and subtitle.
20. **\_createAxis：**Like the name, create axises, include valueAxis and labelAxis.
21. **\_createBackground：**Create background, include border and background pure color or gadient color.
22. **\_createLegend：**Create legend.
23. **\_createScales：**Create scales.
24. **\_createTip：**Create single tip.
25. **\_createTitle：**Create title and subtitle.
26. **\_createAssists：**Create peripheral region include title, legend, axises, beyond the region for animation.
27. **\_formatAxisData：**Unified function to format data for axises. This function is only called in graphics that need to draw axises, such as Area, Bar, Line, Points, etc, because data in these graphic is quite complicated. On the other hand, data in Pie or similar graphic is simple, so there’s no need to organize unified function to format data.
28. **\_getCheckOptions：**This function is to get configs that are used to check options in specific graphic. This function is defined in each graphic and called by \_checkOptions function in core js.
29. **\_getDrawableCoordinate：**Get coordinate information of drawable part of canvas. The drawable part is used for animation.
30. **\_getFormatDiff：**To get difference between the two values according to the data type. There are three parameters: valueType, small, big. When valueType is ‘d’, function return count of days that two datetime value differ, when valueType is ‘t’, return count of minutes that two datetime value differ.
31. **\_getMouseLoction：**Get the mouse position relative to the canvas, return an object include X(left) and Y(top).
32. **\_onStart：**This function is called in Draw function in each graphic to do some “basic works”, like initial core inner objects, such as coordinates, tempData, etc. Built-in event onStart is triggered in this function.
33. **\_resetSharedOpions：**Set defaults back to common options. Differ with SetDefaultOptions function, this function only handles common options, while SetDefaultOptions handles all options, include common options and specific options.
34. **\_spreadSkin：**Spread skin function. SetSkin function is defined in core js, therefore, \_spreadSkin function is defined in each specific graphic and called by SetSkin function in order to make skin really effective.
35. **\_startDrawAndAnimation：**Start drawing. Defined in core js and called by Draw function in each grahpic.
36. **\_getComputed**：Unified function to compute refered min value, max value, interval value, count of lines(means (max-min)/interval). The function compute refered min, max, interval through parameters of valueType, real min, max valve.