

Не баш кратак увод у ConT_EXt Mark IV

Не баш кратак увод у ConTeXt Mark IV

Верзија 1.6 [2. јануар 2021]

© 2020-2021, Joaquín Ataz-López

Наслов оригинала: Una introducción (no demasiado breve) a ConTeXt Mark IV

Превод на српски: добар пријатељ који жели да остане анониман.

Аутор овог текста (и његов преводилац на српски) дозвољава слободну дистрибуцију и употребу, укључујући и право на умножавање и редистрибуцију садржаја овог документа у дигиталном облику под условима да се потврђује ауторство и да се не укључује у било који софтверски пакет или скуп програма, или у документацију чији услови употребе или дистрибуције не укључују слободно право прималаца да је копирају и дистрибуирају. Исто тако, даје се дозвола за превођење овог документа, под условом да се наведе ауторство оригиналног документа и да се преведени текст дистрибуира под FDL лиценцом коју објављује *Free Software Foundation*, *Creative Commons* лиценцом која дозвољава копирање и редистрибуцију, или неком сличном лиценцом.

Уз све ово, издавање или оглашавање или превођење овог документа у папирном облику захтева писмену личну ауторову дозволу.

Историја верзија:

- 18. август 2020: Верзија 1.0 (само на шпанском): оригинални документ.
- 23. август 2020: Верзија 1.1 (само на шпанском): исправка мањих грешака, ауторових грешака у куцању и неразумевања.
- 3. септембар 2020: Верзија 1.15 (само на шпанском): још грешака, грешака у куцању и неразумевања.
- 5. септембар 2020: Верзија 1.16 (само на шпанском): још грешака, грешака у куцању и неразумевања, као и неке врло ситне измене које (надам се) чине текст јаснијим.
- 6. септембар 2020: Верзија 1.17 (само на шпанском): број ситних грешака које проналазим је невероватан. Једноставно би требало да престанем са поновним читањем ако желим да их више не проналазим!
- 21. октобар 2020: Верзија 1.5 (само на шпанском): унос сугестија и исправки грешака које су пријавили корисници NTG-context мејлинг листе.
- 2. јануар 2021: Верзија 1.6: исправке које су предложене након пажљивог читања документа, приликом његовог превођења на енглески језик. Ово је прва верзија на енглеском.

Садржај

| | |
|--|-----------|
| Предговор | 7 |
| I Шта је ConT_EXt и како се користи | 12 |
| 1 ConT_EXt: општи преглед | 13 |
| 1.1 Па шта је уопште ConT _E Xt? | 13 |
| 1.2 Словослагање текстова | 14 |
| 1.3 Језици за означавање | 15 |
| 1.4 T _E X и језици изведени из њега | 15 |
| 1.5 ConT _E Xt | 17 |
| 2 Наш први изворни фајл | 23 |
| 2.1 Припрема експеримента: неопходни алати | 23 |
| 2.2 Сам експеримент | 24 |
| 2.3 Структура фајла нашег примера | 28 |
| 2.4 Неки додатни детаљи у вези покретања команде „context” | 28 |
| 2.5 Управљање грешкама | 29 |
| 3 Команде и остали фундаментални концепти система ConT_EXt | 32 |
| 3.1 ConT _E Xt резервисани карактери | 32 |
| 3.2 Саме команде | 35 |
| 3.3 Опсег важења команди | 37 |
| 3.4 Опције рада команде | 39 |
| 3.5 Резиме синтаксе команде и опција, употреба великих и витичастих заграда приликом позивања | 41 |
| 3.6 Званична листа ConT _E Xt команди | 42 |
| 3.7 Дефинисање нових команди | 42 |
| 3.8 Остали основни концепти | 45 |
| 3.9 Метода за самостално учење система ConT _E Xt | 48 |
| 4 Изворни фајлови и пројекти | 50 |

| | | |
|-----|--|----|
| 4.1 | Кодирање изворних фајлова | 50 |
| 4.2 | Карактери у изворном фајлу које ConTeXt третира на посебан начин | 52 |
| 4.3 | Прости пројекти и пројекти са више фајлова | 54 |
| 4.4 | Структура изворног фајла у простим пројектима | 54 |
| 4.5 | Рад са више фајлова у T _E X стилу | 55 |
| 4.6 | ConTeXt пројекти у ужем смислу | 57 |

II Глобални аспекти документа 61

5 Странице и пагинација документа 62

| | | |
|-----|--|----|
| 5.1 | Величина странице | 62 |
| 5.2 | Елементи на страници | 65 |
| 5.3 | Распоред странице (<code>\setuplayout</code>) | 66 |
| 5.4 | Нумерација страница | 70 |
| 5.5 | Форсирани или предложени преломи странице | 72 |
| 5.6 | Заглавља и подножја | 73 |
| 5.7 | Уметање текст елемената у ивице и маргине странице | 75 |

6 Фонтови и боје у систему ConTeXt 78

| | | |
|-----|---|----|
| 6.1 | Типографски фонтови укључени у „ConTeXt Standalone” | 78 |
| 6.2 | Особине фонта | 79 |
| 6.3 | Постављање главног фонта документа | 80 |
| 6.4 | Промена неких особина фонта | 81 |
| 6.5 | Остале ствари везане за употребу неких алтернатива | 84 |
| 6.6 | Употреба и конфигурација боја | 84 |

7 Структура документа 87

| | | |
|-----|--|----|
| 7.1 | Структурна подела у докуменатима | 87 |
| 7.2 | Типови одељака и њихова хијерархија | 87 |
| 7.3 | Заједничка синтакса команди поделе | 88 |
| 7.4 | Формат и конфигурација одељака и њихових наслова | 88 |
| 7.5 | Дефинисање нових команди поделе | 91 |
| 7.6 | Макроструктура документа | 92 |

8 Садржаји, индекси, листе 93

| | | |
|-----|---|----|
| 8.1 | Садржај | 93 |
| 8.2 | Листе, комбиноване листе и садржаји базирани на листи | 96 |
| 8.3 | Индекс | 98 |

9 Референце и хиперлинкови 100

| | | |
|-----|---|-----|
| 9.1 | Типови референци | 100 |
| 9.2 | Интерне референце | 100 |
| 9.3 | Интерактивни електронски документи | 103 |
| 9.4 | Хиперлинкови на спољашње документе | 103 |
| 9.5 | Креирање маркера у финалном PDF фајлу | 105 |

III Специфична питања 106

10 Карактери, речи, текст и хоризонтални размак 107

| | | |
|------|---|-----|
| 10.1 | Добијање карактера којима нормално не може да се приступи са тастатуре | 107 |
| 10.2 | Специјално форматирање карактера | 110 |
| 10.3 | Размак између карактера и речи | 112 |
| 10.4 | Сложенице | 113 |
| 10.5 | Језик текста | 113 |

11 Пасуси, линије и вертикални размак 117

| | | |
|------|---|-----|
| 11.1 | Пасуси и њихове карактеристике | 117 |
| 11.2 | Вертикални размак између пасуса | 118 |
| 11.3 | Како ConTeXt изграђује линије које формирају пасусе | 119 |
| 11.4 | Проред | 121 |
| 11.5 | Остале ствари у вези линија | 121 |
| 11.6 | Хоризонтално и вертикално поравнање | 122 |

12 Специјалне конструкције и пасуси 124

| | | |
|------|---|-----|
| 12.1 | Фусноте и белешке на крају | 124 |
| 12.2 | Пасуси у више колона | 127 |
| 12.3 | Уређене листе | 129 |
| 12.4 | Описи и набрајања | 131 |
| 12.5 | Линије и оквири | 133 |
| 12.6 | Остала интересантна окружења и конструкције | 134 |

13 Слике, табеле и остали плутајући објекти 136

| | | |
|------|---|-----|
| 13.1 | Шта су плутајући објекти и шта они раде? | 136 |
| 13.2 | Спољне слике | 136 |
| 13.3 | Табеле | 139 |
| 13.4 | Заједнички аспекти за слике, табеле и остале плутајуће објекте .. | 141 |
| 13.5 | Дефинисање додатних плутајућих објеката | 143 |

Додаци 145

| | | |
|----------|---|------------|
| A | Инсталација, конфигурација и ажурирање система | |
| | ConT_EXt | 146 |
| 1 | Инсталирање и конфигурисање „ConT _E Xt Standalone” | 146 |
| 2 | Инсталација LMTX | 148 |
| 3 | Употреба неколико верзија система ConT _E Xt на истом систему (само за Unix системе) | 149 |
| Б | Команде за генерисање математичких и осталих симбола | 150 |
| В | Индекс команди | 152 |

Предговор*

Поштовани читаоче, овај документ описује ConT_EXt, словослагачки систем изведен из система T_EX, такође словослагачког система, који је између 1977. и 1982. године креирао Доналд Е. Кнут при универзитету Стенфорд.

ConT_EXt је осмишљен за креирање докумената врло високог типографског квалитета – било папирних докумената или оних предвиђених да се приказују на екрану рачунара. То није текст процесор или текст едитор, али, као што сам раније поменуо, *систем*, или другим речима скуп алата намењен за словослагање докумената, што представља графички распоред и визуелизацију различитих елемената документа на страници или на екрану. Укратко, ConT_EXt тежи да обезбеди све алате неопходне да документи добију најбољи могући изглед. Идеја је да будете у могућности да генеришете документе који, осим тога што су лепо написани, такође и лепо „изгледају”. У том смислу, овде можемо поменути шта је Доналд Е. Кнут написао када је представљао T_EX (систем на којем је базиран ConT_EXt):

Ако само желите да направите документ довољно добар да прође—нешто прихватљиво и у основи читљиво, али не и заиста лепо—обично ће вам бити довољан једноставнији систем. Циљ система T_EX је да произведе најфинији квалитет; а то захтева више пажње на детаље, али вам неће бити много теже да дођете до тог циља, а бићете и посебно поносни на завршени производ.

Када припремамо рукопис системом ConT_EXt, прецизно наводимо како он треба да се трансформише у странице (или екране) чији је типографски квалитет може да се упоређи са оним што могу произвести најбоље штампарије на свету. Да бисмо ово урадили, једном када научимо систем, потребно нам је само мало више рада у односу на онај потребан да се откуца документ у било ком текст процесору или текст едитору. Уствари, када једном достигнумо одговарајућу лакоћу у руковању системом ConT_EXt, укупан посао је вероватно мањи ако имамо на уму да се у систему ConT_EXt већина детаља формирања документа описује глобално и ради се са текст фајловима који су – једном када се навикнемо на њих – много природнији начин за рад на креирању и уређивању докумената; а чињеница је и да је ова врста фајлова много мања и једноставнија за обраду од комплексних бинарних фајлова које користе текст процесори.

Постоји поприлична количина документације о систему ConT_EXt, и скоро сва је на енглеском језику. Оно што бисмо могли сматрати да је *званична* дистрибуција система ConT_EXt – под називом „ConT_EXt Standalone”² – на пример, садржи неких 180 PDF фајлова документације (од којих је већина на енглеском језику, али су неки и на холандском и немачком) укључујући упутства, примере и техничке чланке; а на Pragma ADE веб сајту (компанија која је изнедрила ConT_EXt) постоји (на дан када сам пребројао у мају 2020. године) 224 документа које можете преузети, од којих се већина дистрибуира уз „ConT_EXt Standalone” али и неких других. Свакако, ова огромна документација није

* Овај предговор је започет са намером да буде превод/адаптација на ConT_EXt предговора књиге „The T_EXBook”, документа који објашњава *све што је потребно да знате о програму T_EX*. На крају сам морао да одступим од тога; ипак, задржао сам неке фрагменте који ће онима који га познају, надам се, пружити неке његове одјекве.

² У време када је писана прва верзија овог текста, оно што је овде речено је било поткрепљено чињеницама; али у пролеће 2020 године, ConT_EXt вики је ажуриран и од тада морамо да претпоставимо да је „званична” дистрибуција система ConT_EXt постала LMTX. Ипак, за оне који по први пут улазе у ConT_EXt свет, још увек бих препоручио да користе „ConT_EXt Standalone” јер је то стабилнија дистрибуција. **Додатак А** објашњава како да инсталирате било коју од ових дистрибуција.

заиста од користи за учење система ConTeXt јер, у општем случају, ови документи нису намењени читаоцу који не зна ништа у вези система, али жели да га научи. Од 56 PDF фајлова које „ConTeXt Standalone” назива „упутства”, постоји само један који претпоставља да читалац не зна ништа у вези система ConTeXt. То је документ под именом „ConTeXt Mark IV, an Excursion”. Међутим, овај документ, као што му само име говори, ограничава свој садржај на представљање система и објашњавање начина на који се раде одређене ствари које могу да се ураде системом ConTeXt. Он би био добар увод ако би се наставио мало боље уређеним и систематским референтним приручником. Такво упутство не постоји, па је празнина између ConTeXt „Excursion” и остатка документације сувише велика.

2001. године је написан референтни приручник који може да се пронађе на [Pragma ADE веб сајту](#); али упркос наслову, с једне стране он није дизајниран тако да буде *комплетно упутство*, док је с друге стране био (и јесте) текст намењен претходној верзији система ConTeXt (под називом Mark II), па је стога прилично застарео.

Упутство је 2013. године делимично ажурирано, али многи његови одељци нису поново написани тако да садржи информације које се тичу и система ConTeXt Mark II и ConTeXt Mark IV (текуће верзије), без потпуно јасног назначивања која информације се односи на сваку од верзија. Вероватно је ово разлог што се ово упутство не налази међу документима који су део „ConTeXt Standalone”. Без обзира на ове мане, упутство и даље представља најбољи документ за почетак учења система ConTeXt онда када се прочита уводни „ConTeXt Mark IV, an Excursion”. За прве кораке у систему ConTeXt такође су корисне информације које могу да се пронађу на његовом [вики](#) веб сајту који се, у време писања овог текста, редизајнира тако да добија много јаснију структуру, мада и он меша објашњења која функционишу само у Mark II са осталима за Mark IV или за обе верзије. Овај недостатак диференцијације се такође проналази у званичној листи ConTeXt команди¹ која не наводи које команде раде само у једној од две верзије.

У основи, овај увод је написан узимајући информације из четири извора која се овде наводе: упутство ConTeXt „Excursion”, упутство из 2013. године, садржај вики веб сајта и званична листа команди у којој се налази, за сваку од њих, допуштена конфигурација опција; уз то, наравно, и моји сопствени тестови и закључци. Дакле, овај увод у суштини представља резултат истраживачког рада, па сам неко време био размишљао да ли да га назовем „Шта знам о ConTeXt Mark IV” или „Шта сам научио о ConTeXt Mark IV”. Коначно, одбацио сам ове наслове јер, колико год да су истинити, осећао сам да бих њима могао одвратити некога од упознавања са системом ConTeXt; и оно што је сигурно је то да мада документација има (према мом мишљењу) неке недостатке, имамо заиста користан и свестран алат за који се труд потребан да се научи несумњиво исплати. Користећи ConTeXt можемо да манипулишемо и конфигуришемо текст документа тако да постигнемо ствари које неко ко не познаје систем не може ни да замисли.

Због своје личне природе, не могу спречити да се у овом документу с времена на време појаве притужбе у вези недостатка информација. Не бих желео да се ово схвати погрешно: ја сам веома захвалан творцима система ConTeXt што су дизајнирали тако моћан алат и учинили га јавно доступним. Једноставно не могу да се одупрем мишљењу да би овај алат био много популарнији ако би се унапредила његова документација: потребно је да уложи много времена да се система научи, не због његове сложености (која је неоспорна, али није већа од неких сличних алата – уствари чак напротив), већ услед недостатка јасних, комплетних и добро организованих информација које праве разлику између две верзије система ConTeXt, објашњавајући функције у свакој од њих, и првенствено, јасно наводећи шта свака команда, аргумент и опција раде.

Тачно је да ова врста информација захтева улагање доста времена. Али ако се има у виду да многе команде деле опције са сличним именима, можда би могла да се обезбеди нека врста *речника* опција што би такође помогло да се открију неке недоследности које се јављају када две опције са истим именом раде различите ствари, или када је потребно да се за исту ствар у две различите команде користе опције са другачијим именом.

Као читаоца који по први пут приступа систему ConTeXt, нека вас моје притужбе не одврате од учења система, јер мада може бити тачно да недостатак информација продужава време потребно за учење система, бар што се тиче материјала о коме се говори у овом уводу, ја сам већ уложио то време тако да читалац нема потребе то да ради. И само са оним што научи из овог увода, читаоци ће на располагању имати алат који им омогућава да једноставно произведу документе какве никада нису очекивали да ће бити у стању да произведу.

Пошто оно што је објашњено у овом документу у највећој мери долази из мојих сопствених закључака, врло је могуће да, упркос томе што сам лично тестирао већину онога о чему говорим, неке изјаве

¹ За листу погледајте [одељак 3.6](#).



или мишљења можда нису у потпуности исправна нити уобичајена. Ценићу, наравно, било какву исправку, побољшање или разјашњење које читаоци могу да ми понуде, и оне могу да се пошаљу на адресу joaquin@ataz.org. Међутим, да би се смањило број ситуација у којима је вероватно да нисам у праву, покушао сам да не залазим у материју о којој нисам пронашао никакве информације и коју нисам могао (или желео) лично да испробам. Понекад је ово случај јер резултати мојих тестова нису били убедљиви, а понекад јер нисам успео све да тестирам: број команди и опција које има систем ConTeXt је импресиван, па ако бих морао све да испробам, никада не бих завршио овај увод. Међутим, постоје прилике у којима не могу да избегнем *претпостављање* нечега, нпр. исказ да нешто видим као вероватно, али да нисам у потпуности сигуран у то. У тим случајевима је на леву маргину пасуса у којем наводим такву претпоставку постављена слика 'нагађања'. Сврха слике је да графички прикаже претпоставку.¹ У другим ситуацијама нисам имао избора осим да признам како нешто не знам и да немам разумну претпоставку у вези тога: у том случају, слика која се приказује непосредно лево у маргини служи да покаже више од нагађања или незнања.² Али пошто никада нисам био добар са графичким представљањем, нисам сигуран да изабране слике успевају да пренесу толико финеса.

С друге стране, овај увод је написан из угла читаоца који не зна ништа о системима T_EX или ConT_EXt, мада се надам да такође може бити користан и неке ко долази са система T_EX или L^AT_EX (најпопуларнијим од система изведених из T_EX) и који по први пут приступају систему ConT_EXt. Исто тако, свестан сам да тиме што покушавам да задовољим сваког читаоца, постоји ризик да нико не буде задовољан. Стога, у случају да постоји сумња, увек сам јасно стављао до знања да је књига углавном намењена неке ко се по први пут упознаје са системом ConT_EXt, неке ко је тек наишао на овај очаравајући екосистем.

То што је неко почетник у систему ConT_EXt не повлачи да је он такође и почетник у употреби компјутерских алата; и мада у овом уводу не претпостављам да читаоци поседују било какав одређени ниво компјутерске писмености, ипак претпостављам одређену „разумну писменост“ која на пример значи да се поседује опште разумевање разлике између текст процесора и текст едитора, познавање начина да се креира, отвори и манипулише текст фајл, познавање начина да се инсталира програм, познавање начина да се отвори терминал и изврши команда... и још по нешто.

Читајући претходне делове увода док пишем ове линије, схватам да се понекад занесем у компјутерске проблеме који нису неопходни за учење система ConT_EXt и да би то могло да уплаши почетника, док сам у другим ситуацијама заузет објашњавањем прилично очигледних ствари које би могле бити досадне искусном читаоцу. Молим за попустљивост и од једног и од другог. Наравно да разумем како је почетнику у компјутеризованом управљању текстом веома тешко да уопште зна за постојање система ConT_EXt, али ја сам у свом професионалном окружењу окружен људима који се константно боре са текстовима док користе текст процесоре и то чине прилично успешно, али пошто никада нису радили са текст фајловима, они игноришу неке основне ствари као што су, на пример, кодирање које се користи у фајлу или шта је разлика између текст процесора и текст едитора.

Чињеница да је овај приручник дизајниран за људе који не знају ништа о систему ConT_EXt или T_EX, повлачи да сам навео и информације које очигледно нису везане за ConT_EXt већ за T_EX; али сам схватио да није неопходно оптеретити читаоце информацијама које им нису битне, што би могао да буде случај када нека команда која у *суштини* функционише, заиста ConT_EXt команда, или припада систему T_EX; тако да само у неким ситуацијама, када ми се чини да је то корисно, разјашњавам да одређена команда уствари припада систему T_EX.

У погледу организације овог документа, материјал је груписан у три блока:

- **Први део**, који се састоји од прва четири поглавља, даје уопштени преглед система ConT_EXt, објашњава шта је он и како се ради са њим, приказује први пример трансформисања документа тако да би касније могли да се објасне неки фундаментални концепти система ConT_EXt заједно са одређеним питањима у вези ConT_EXt изворних фајлова.

Као целина, ова поглавља су намењена читаоцима који до сада познају само рад са текст процесорима. Читалац који већ зна да ради са означавајућим језицима би могао да прескочи ова рана

¹ Сliku нисам ја нацртао, већ сам је преузео са интернета (<https://es.dreamstime.com/>), где се каже да је то бесплатна слика.

² Такође је пронађена на интернету (<https://www.freepik.es/>) где се одобрава слободна употреба.

поглавља; а ако читалац већ познаје \TeX или \LaTeX , могли би такође да прескоче и већину садржаја поглавља 3 и 4. Исто тако, препоручио бих да се прочита барем:

- Информације у вези \ConTeXt команди (поглавље 3), а поготово начин на који команда функционише, како се конфигурише, јер се овде налази основна разлика између концепције и синтаксе система \LaTeX и \ConTeXt . Пошто се овај увод односи само на овај други, ове разлике се не изражавају директно, али неко ко чита ово поглавље и зна начин на који функционише \LaTeX ће одмах разумети разлику у синтакси ова два језика, а као и начин на који нам \ConTeXt дозвољава да конфигуришемо и прилагодимо начин на који раде скоро све његове команде.
 - Информације о \ConTeXt пројектима са више фајлова (поглавље 4), што се прилично разликује од начина рада са осталим системима заснованим на систему \TeX .
 - **Други део**, који чине поглавља 5 до 9, фокусира се на оно сматрамо да су основни општи аспекти \ConTeXt документа:
 - Два аспекта која углавном утичу на изглед документа су величина и распоред његових страна, као и фонт који се користи. Поголавља 5 и 6 су посвећена овим стварима.
 - * Прво се посвећује страницама: величином, елементима који чине страницу, распоред тих елемената (што значи начин на који су распоређени елементи) итд. Из разлога систематизације, овде се говори и о одређенијим аспектима као што су они који се тичу нумерације страница и механизма који омогућавају да се на утиче на нумерацију.
 - * Поголавље 6 објашњава команде у вези фонтова и управљања фонтовима. Ту се такође налази основно упутство за употребу и управљање бојама, мада оне нису стриктно *карактеристика* фонтова, ипак у истој мери утичу на спољашњи изглед документа.
 - Поголавља 7 и 8 се баве структуром документа и алатима које \ConTeXt нуди ауторима као помоћ у писању добро структурираних докумената. Поголавље 7 се фокусира на структуру документа у ужем смислу (структурну поделу документа), а поглавље 8 на начин којим се та структура огледа у садржају; мада уз ово објашњење користимо прилику да такође објаснимо како се генеришу различите врсте индекса системом \ConTeXt , јер све то у систему \ConTeXt потпада под појам „листи“.
 - Коначно, поглавље 9 се бави референцама, важном глобалном аспекту било ког документа онда када је потребно да укажемо на нешто шта се налази у неком другом делу документа (интерне референце) или у другим документима (спољашње референце). У случају ових других, тренутно нас интересују референце (линкови) који воде на спољашњи документ. Ови *линкови* (који такође могу да се јаве и у интерним референцама) чине да наш документ буде *интерактиван*, и у овом поглављу показујемо неке од могућности система \ConTeXt које служе за креирање таквих докумената.
- Није потребно да се ова поглавља читају у било ком одређеном редоследу, осим поглавље 8, које би могло лакше да се разуме ако се прво прочита поглавље 7. У сваком случају, покушао сам да осигурам да када у поглављу или одељку искрсне питање о којем се говори на неком другом месту у овом уводу, текст садржи опаску о томе заједно са хиперлинком на место где се питање обрађује. Ипак, нисам у позицији да гарантујем како ће ово увек бити случај.
- Коначно, **трећи део** (поглавља 10 и наредна) се фокусира на детаљније аспекте. Они су независни, не само једни од других, већ чак и од својих одељака (осим можда у последњем поглављу). Имајући у виду велики број алата који су део система \ConTeXt , овај део би требало да буде прилично опширан; али како сматрам да док читаоци они стигну овде, већ ће бити припремљени да се сами уроне у \ConTeXt документацију, укључио сам само следећа поглавља:
 - Поголавља 10 и 11 се баве оним што бисмо могли да назовемо *основни елементи* било ког текст документа: текст је исписан карактерима који чине речи које су груписане у линије,

које са своје стране чине пасусе раздвојене један од другог вертикалним простором... Јасно је да би све ове ствари могле да се сместе у једно поглавље, али пошто би оно било предугачко, материју сам поделио у два поглавља, једно које се бави карактерима, речима и хоризонталним размацама, а друго које се бави линијама, пасусима и вертикалним размацама.

- Поглавље 12 је нека врста *мешавине* која се бави елементима и конструкцијама које се обично срећу у документима; углавном у академским и техничким документима; највећим делом у академским, научним или техничким документима: фусноте, структурне листе, набрајања, итд.
- Коначно, поглавље 13 се фокусира на пливајуће објекте, посебно оне који се најчешће користе: слике које се умећу у документе и табеле.
- Увод се завршава са три **додатка**. Један је у вези инсталације система ConT_EXt, а други додаток садржи неколико десетина команди које генеришу различите симболе - углавном, мада не само за математичку употребу и трећи садржи азбучну листу ConT_EXt команди које су поменуте или објашњене раније у тексту.

Постоји много тема које још увек треба да се објасне: рад са цитатима и библиографским референцама, писање специјализованих текстова (математичких, хемијских...), веза са XML, интерфејс са Lua кодом, режими и обрада заснована на режимима, рад са MetaPost језиком за дизајнирање графике, итд. То је разлог због кога сам, како не дајем комплетно објашњење система ConT_EXt, нити се правим да то чиним, назвао овај документ „Увод у ConT_EXt Mark IV”; и додао сам чињеницу да увод није баш кратак, јер очигледно је тако: текст који је оставио још доста тога недовршеног, али је већ стигао до више од 300 страница никако не може да се назове кратак увод. Желим да читалац разуме логику система ConT_EXt, или барем онако како сам је ја разумео. Не покушава да се представи као референтно упутство, већ пре водич за самостално учење који припрема читаоца да прави документе средње сложености (а то значи већину вероватних докумената) и да првенствено учи читаоца да *замисли* шта може да се уради овим моћним алатом, као и да у доступној документацији сазна како то да изврши. Овај документ није ни *туторијал*. Туторијали су дизајнирани тако да прогресивно повећавају ниво сложености, тако да се корак по корак прелази оно што треба да се научи; када се ово има на уму, ја сам хтео да почнем са другим делом уместо да поређам материјал по степену сложености, тако да будем систематичнији. Али како ово није туторијал, представио сам велики број примера.

Можда ће наслов овог документа неке читаоце да подсећа на текст доступан на интернету који су написали Етикер, Партл, Хина и Шлегл, један од бољих докумената који читаоца уводи у L^AT_EX свет. Мислим на „*The Not So Short Introduction to L^AT_EX 2_ε*”. То није случајност, већ одавање почasti и уважавање: захваљујући онима који пишу текстове као што је овај, многи људи могу почети да раде са корисним и моћним алатима као што су L^AT_EX и ConT_EXt. Ови аутори су ми помогли да почнем са системом L^AT_EX; ја се надам да могу урадити исто за некога ко жели да почне са системом ConT_EXt, мада сам оригиналну шпанску верзију текста наменио само делу света који говори шпански језик и којем недостаје много документације на њиховом језику. Надам се да овај документ испуњава очекивања и у међувремену, други су несебично понудили помоћ да се он преведе на остале језике. Отуда и ово енглеско издање. Хвала вам.

Хоакин Атас-Лопес
Лето 2020

I

Шта је ConT_EXt и како се користи

Глава 1

ConTeXt: општи преглед

Садржај: 1.1 Па шта је уопште ConTeXt?; 1.2 Словослагање текстова; 1.3 Језици за означавање; 1.4 TeX и језици изведени из њега; 1.4.1 TeX машине; 1.4.2 Формати који су изведени из TeX; 1.5 ConTeXt; 1.5.1 Кратка историја система ConTeXt; 1.5.2 ConTeXt наспрам LaTeX; 1.5.3 Добро разумевање динамике рада у систему ConTeXt; 1.5.4 Добијање помоћи у вези система ConTeXt;

1.1 Па шта је уопште ConTeXt?

ConTeXt је *словослагачки систем*, или другим речима: опсежан скуп алата креиран тако да кориснику пружи апсолутну и потпуну контролу над изгледом и презентацијом специфичних електронских докумената намењених за штампање на папиру или за приказ на екрану. Ово поглавље објашњава шта то све значи. Али најпре, хајде да истакнемо неке од карактеристика система ConTeXt.

- Постоје две *врсте* система ConTeXt познате као Mark II и Mark IV. ConTeXt Mark II је замрзнут, тј. сматра се да је потпуно развијен језик који се убудуће неће мењати, нити ће добијати нове могућности. Нова верзија би се појавила само у случају да постоји грешка која мора да се исправи. С друге стране, ConTeXt Mark IV наставља да се развија и с времена на време се појављују нове верзије које доносе нека побољшања или додатне могућности. Али, мада се још увек развија, он је прилично зрео језик у који нове верзије уносе прилично суптилне измене које скори искључиво утичу на функционисање система на ниском нивоу. За обичног корисника су ове измене потпуно транспарентне; као да се уопште нису ни унеле. Мада обе *врсте* имају доста тога заједничког, оне имају и неке некомпатибилне могућности. Из тог разлога се овај увод фокусира само на ConTeXt Mark IV.
- ConTeXt је софтвер *libre* (или слободан софтвер, али не у смислу да је *бесплатан*). Тачније, програм (односно комплекс компјутерских алата који чине ConTeXt), се дистрибуира под *ГНУ Општом Јавном Лиценцом*. Документација се доставља под „Creative Commons” лиценцом која дозвољава да се слободно копира и дистрибуира.
- ConTeXt није ни текст процесор ни програм за уређивање текста, већ колекција алата намењених *трансформисању* текста који је написан омиљеним текст едитором. Стога, када радимо са системом ConTeXt:
 - Почињемо тако што пишемо један или више текст фајлова било којим текст едитором.
 - Ови фајлови, заједно са текстом који чини садржај документа имају низ инструкција које систему ConTeXt говоре о изгледу који мора да има финални документ генерисан из оригиналних текст фајлова. Уствари, комплетан скуп ConTeXt инструкција је *језик*; и пошто овај језик омогућава *програмирање* типографске трансформације текста, можемо рећи да је ConTeXt *типograфски програмски језик*.
 - Када напишемо изворне фајлове, програм (који се такође зове „context”¹) ће од њих да генерише PDF фајл који је спреман за слање у штампарију или приказ на екран.

¹ ConTeXt је у исто време и језик и програм (а и још неке друге ствари). У тексту као што је овај, та чињеница прави проблем, јер понекад морамо направити разлику између ова два аспекта. Због тога сам усвојио типографску конвенцију да када говорим о језику „ConTeXt”, или о језику и о програму, његово име пишем користећи логотип (ConTeXt). Међутим, када желим да говорим само о програму, онда име пишем „context” користећи сва мала слова и фонт фиксне ширине, типичан за компјутерске термине и писаће машине. Овај фонт ћу такође да користим и за примере и помињања команди које су део овог језика.

- Дакле, у систему ConTeXt морамо да направимо разлику између документа који пишемо и документа који генерише ConTeXt. Да би се спречиле било какве недоумице, документ који садржи инструкције за форматирање ћу у овом уводу да зовем *изворни фајл*, а PDF документ који из изворног фајла генерише ConTeXt ћу да зовем *финални документ*.

О овим основним стварима ћемо још расправљати мало касније.

1.2 Словослагање текстова

Писање документа (књиге, чланка, поглавља, проспекта, рада...) и типографско састављање свега су две потпуно различите активности. Писање документа је скоро исто као писање оловком; то ради аутор који одлучује о његовом садржају и структури. Документ који прави директно аутор, онако као да га је он или она написао, назива се *рукопис*. Природно је да само аутор, или они који имају право да га читају, могу да приступе рукопису. Дељење рукописа ван ове мале групе захтева да се рукопис *објави*. Данас је то – у етимолошком смислу омогућавања „јавног приступа” – просто као постављање на интернет, тако да је рукопис доступан сваком ко га пронађе и жели да га прочита. Али све до релативно скоро, објављивање је било прилично скуп процес који зависи од одређених професионалаца специјализованих за то, вољних да приступе рукопису који сматрају довољно значајним, било због његовог садржаја, било због његових аутора. Па чак и данас тежимо да реч *публикација* резервишемо за ову врсту *професионалних издања* код којих рукопис пролази кроз низ трансформација свог изгледа чији циљ је да се унапреди *читљивост* документа. Овај низ трансформација је оно што називамо *словослагањем*.

Циљ словослагања је – у општем случају, остављајући по страни текстове маркетиншког типа који покушавају да привуку пажњу читаоца – да произведе документе највеће *читљивости*, што подразумева квалитет штампаног текста који позива на читање или га омогућава, и осигурава да је читаоцу удобно да чита. Овоме доприноси много ствари; неке се, наравно, тичу *садржаја* документа: (квалитет, јасноћа, организација...), али остале зависе од ствари као што су врста и величина употребљеног фонта, употреба празног простора у документу, визуелно раздвајање пасуса, итд. Уз то, постоје и друге врсте ресурса које нису толико графичког или визуелног типа, као што су присуство одређених помоћних средстава читаоцу – заглавља и подножја страница, индекси, речници, употреба црног слога, наслова у маргини, итд. Знање и исправна употреба свих ресурса који су доступни словослагачу би могло да се назове „уметност словослагања” или „уметност штампања”.

У прошлости, све до појаве компјутера, задаци и улоге писца и словослагача су били прилично раздвојени. Аутор је писао руком, или у XIX веку писаћом машином чији су типографски ресурси су били ограничени, чак и више него онима који су ручно писали; тада је писац прослеђивао оригинале издавачу или штампару, који их је трансформисао тако да се добије штампани документ.

Данас је компјутерска наука аутору олакшала одлучивање о композицији све до најситнијих детаља. Међутим, то не умањује важност чињенице да квалитети потребни за доброг аутора нису исти као они који су потребни за доброг словослагача. Зависно од врсте документа, аутору је потребно разумевање материје о којој пише, јасноћа излагања, добро организовано размишљање које води до добро организованог текста, креативност, осећај за ритам, итд. Али словослагач мора да комбинује добро знање концептуалних и графичких ресурса који су му на располагању, и довољно доброг укуса како би могао складно да их употреби.

Добрим програмом за обраду текста¹ је могуће постићи типографски разумно добро припремљен документ. Али текст процесори, у општем случају, нису дизајнирани за словослагање, па резултати, мада могу бити коректни, не могу да се пореде са резултатима који се добију помоћу других алата дизајнираних за контролу композиције документа. Уствари, текст процесори су еволуирали из писаћих машина, па њихова употреба, услед тога што маскирају разлику између ауторства текста и

¹ Према прилично старој конвенцији, правимо разлику између *текст едитора* и *текст процесора*. Рани програми за уређивање текста су обрађивали неформатирани текст фајлове, док су друге врсте програма радиле са бинарним фајловима форматираног текста.

словослагања текста, води ка типографски неадекватним текстовима којима недостаје структура. С друге стране, алати као што је ConTeXt су еволуирали из штампарске пресе; они нуде много више композиционих могућности, а изнад свега, начин њихове употребе не може да се научи без успутног овладавања многим појмовима у вези словослагања. То је разлика у односу на текст процесоре које неко може годинама да користи без потребе да научи било коју ствар у вези типографије.

1.3 Језици за означавање

Као што сам већ поменуо, у данима пре компјутера, аутор је рукопис припремао ручно или писаћом машином и предавао га је издавачу или штампару који је био одговоран да га трансформише у финални штампани текст. Мада је аутор сасвим мало био умешан у трансформацију, он или она је утицао на указивање да су, на пример, одређене линије рукописа наслови његових разних делова (поглавља, одељака...), или да одређене ствари треба типографски истаћи на неки начин. Ове ознаке је аутор стављао на сам рукопис, понекад директно, а понекад користећи одређене конвенције које су се временом развијале. На пример, поглавља су увек почињала на новој страници уметањем неколико празних линија испред наслова, подвлачећи наслов, исписујући га верзалом, или постављајући текст који треба да се истакне између две подвлаке, повећавајући увлачење пасуса, итд.

Укратко, аутор је *означавао* текст како би навео начин на који треба да се сложи. Касније би уредник руком исписао у текст остале ознаке намењене штампару, као што су на пример, фонт који треба да се користи и његова величина.

Данас, у компјутеризованом свету, ово настављамо да радимо кад генеришемо електронске документе користећи нешто што се назива *језик за означавање*. Ове врсте језика користе низ *ознака* или индикација које програм за обраду фајлова који их садрже зна како да интерпретира. Тренутно је HTML вероватно најпознатији језик за означавање, јер је већина веб страница базирана на њему. HTML страница садржи текст веб странице заједно са низом ознака које програму за преглед који је читава говорје како би требало да је прикаже. HTML означавање које разумеју веб прегледачи, заједно са инструкцијама о томе где да их користе, се назива „HTML језик“, и то је *језик за означавање*. Али уз HTML постоји много других језика за означавање; они уствари цветају као печурке после кише, тако да се XML, језик за означавање *par excellence*, налази свуда и користи се практично за све: за дизајн база података, за креирање специфичних језика, пренос структурираних података, фајлове конфигурације апликација, итд. Такође постоје језици за означавање намењени графичком дизајну (SVG, TikZ или MetaPost), математичким формулама (MathML), музици (Lilypond и MusicXML), финансијама, геоматици, итд. А такође постоје и језици за означавање који су намењени типографској трансформацији текста, међу којима се истичу T_EX и језици изведени из њега.

Када се говори о *типиграфском* означавању које указује на то како би текст требало да изгледа, постоје две врсте на које можемо да мислимо: *чисто типографско означавање* и *концептуално означавање* или, ако вам тако више одговара, *логичко означавање*. Чисто типографско означавање је ограничено на прецизно указивање типографског ресурса који би требало да се употреби за приказ одређеног текста; као када, на пример, наведемо да би неки текст требало да се испише црним слогом или курзивом. С друге стране, концептуално означавање наводи функцију коју одређени текст има у документу као целини, као када наведемо да је нешто наслов, поднаслов или цитат. У општем случају, документи у којима се користи ова друга врста означавања су конзистентнији и једноставнији за састављање, јер указују на разлику између ауторства и композиције: аутор наводи да је та и та линија наслов, или тај и тај фрагмент упозорење, или цитат; па словослагач одлучује како да типографски истакне све наслове, упозорења или цитате; дакле, с једне стране је гарантована конзистенција, пошто ће сви фрагменти који имају одређену функцију изгледати исто, а са друге стране, тако се штеди време јер је потребно да се формат сваке врсте фрагмента наведе само једном.

1.4 T_EX и језици изведени из њега

T_EX је крајем 70их година развио Доналд Е. Кнут, професор (сада емеритус професор) теоретског компјутерског програмирања на Универзитету Стенфорд. Он је имплементирао програм да би про-

извео сопствене публикације и као пример систематски развијеног и коментарисаног програма. Уз \TeX , Кнут је развио још један програмски језик под именом MetaFont, креиран за дизајнирање типографских фонтова и користио га је да дизајнира фонт који је крстио *Computer Modern*, који, уз уобичајене карактере сваког фонта, такође укључује и комплетан скуп „словних ликова”¹ дизајнираних за писање математике. Свему овоме је додао неколико додатних алата и тако је рођен словослагачки систем под именом \TeX , који се, захваљујући својој снази, квалитету резултата, флексибилности употребе и широким могућностима, сматра за један од најбољих компјутеризованих система за композицију текста. Дизајниран је за текстове у којима има доста математике, али је ускоро било јасно да га системске могућности чине погодним за све врсте текстова.

Интерно, \TeX функционише на исти начин као што су радили стари слагачи у штампаријама. За \TeX , све представља *кутију*: слова се налазе у кутијама, празни простори су такође кутије, неколико слова (кутије које садрже неколико слова) чине нову кутију која садржи реч, а неколико речи, заједно са празним размаком између њих, чини кутију која садржи целу линију, неколико линија постају кутија која садржи пасус... и тако даље. А све ово уз невероватну прецизност при обради раздаљина. Имајте на уму да је најмања јединица коју \TeX може да обради 65.536 пута мања од типографске тачке којом се мере карактери и линије, што је обично и најмања јединица коју могу да обраде већина текст процесор програма. Ово значи да је најмања јединица са којом \TeX може да ради приближно 0,000005356 милиметра.

Име \TeX је изведено из корена грчке речи τέχνη, исписане верзалом (ΤΕΧΝΗ). Стога, последње слово речи \TeX није латинично 'X', већ грчко 'χ', које се изговара као „х”. Дакле, \TeX би требало да се изговара као *tex*. С друге стране, ова грчка реч је значила и „уметност” и „технологија”, па је то разлог што је Кнут баш њу изабрао као име свог система. Сврха овог имена – написао је – „је да вас подсети како се \TeX првенствено тиче техничких рукописа високог квалитета. Његов акценат је на уметности и технологији, као што је то и грчка реч по којој је добио име”.

Користећи конвенцију коју је установио Кнут, \TeX би требало да се напише:

- У типографски форматираним текстовима као што је овај, користећи логотип који сам до сада користио: три слова у верзалу, са средњим 'E' помереним мало наниже, тако да се омогући мањи размак између 'T' и 'X'; или другим речима words: „ \TeX ”.

Да би омогућио писање овог логотипа, Кнут је направио инструкцију у језику \TeX којом се он исписује у финални документ:
 $\backslash\TeX$.

- У неформатираним текстовима (као што је имејл, или текст фајл), са 'T' и 'X' у верзалу, а средњим 'e' у куренту; дакле: „ \TeX ”.

Ова конвенција наставља да се користи у свим системима који се заснивају на систему \TeX тако да укључе своје исправно име, као што је случај са системом ConTeXt. Када се пише у текст режиму требало би да напишемо „ConTeXt”.

1.4.1 \TeX машине

Програм \TeX је слободан *libre* софтвер: његов изворни код је јавно доступан и свако може да га користи или мења како год жели, само уз услов да ако се направе измене, резултат више не сме да се назива „ \TeX ”. Ово је разлог што су се током времена појавиле извесне адаптације програма које су у њега уводиле различита побољшања, а која се уопштено називају *\TeX машине*. Уз оригинални \TeX програм, главне машине су, по хронолошком редоследу појављивања, pdf \TeX , ϵ - \TeX , $\text{X}\text{\TeX}$ и Lua \TeX . Свака од њих би требало да уводи побољшања у односу на претходну. С друге стране, ова побољшања, све до појаве Lua \TeX машине, нису утицала на сам језик, већ само на улазне фајлове, излазне фајлове, управљање изворима и рад макроа на ниском нивоу.

¹ У типографији, словни лик је графичка интерпретација карактера, већег броја карактера, или дела неког карактера, и то је савремени еквивалент реза (ствари у коју је урезано слово или покретно слово).

Питање употребе T_EX машине је једно од питања о којем се у T_EX свету највише расправља. Ово питање нећу да разматрам овде јер ConTeXt Mark IV функционише само са LuaT_EX. У суштини, у ConTeXt свету, дискусија о T_EX машинама постаје дискусија о томе да ли да се користи Mark II (који ради са PdfT_EX и XeT_EX) или Mark IV (који ради само са LuaT_EX).

1.4.2 Формати који су изведени из T_EX

Језгро или срце система T_EX разуме само скуп од око 300 веома основних инструкција, које се називају *примитиве*, и које су погодне за словослагачке операције и функције програмирања. Већина ових функција су веома *ниског нивоа*, што у компјутерској терминологији значи да их компјутери лакше разумеју него људи, јер се тичу елементарних операција типа „помери овај карактер 0,000725 милиметара навише”. Тако да је Кнут увидео да би T_EX требало да буде проширив, што значи да би требало да постоји механизам који омогућава да се дефинишу инструкције на вишем нивоу, инструкције које људи лакше разумеју. Ове инструкције, које се у време извршавања разбијају на једноставније инструкције, називају се *макрои*. На пример, T_EX инструкција која штампа (`\TeX`) логотип, се у време извршавања разлаже као што следи:

```
T
\kern -.1667em
\lower .5ex
\hbox {E}
\kern -.125em
X
```

Али за људско биће је много једноставније да разуме и упамти како једноставна команда „`\TeX`” извршава типографске операције неопходне да се испише логотип.

Разлика између онога што је *макро* и онога шта је *примитива*, у стварности је битна само из перспективе T_EX програмера. Из перспективе корисника, оне су *инструкције* или, ако вам више одговара, *команде*. Кнут их је називао *контролни низови*.

Ова могућност проширивања језика помоћу *макроа* је једна од карактеристика које су учиниле T_EX тако моћним алатом. У суштини, сам Кнут је сачинио око 600 макроа који, заједно са 300 примитива, чине формат који се назива „Plain T_EX”. Врло је уобичајено да се T_EX замени са Plain T_EX, а заправо се све што је написано или речено о T_EX, уствари односи на Plain T_EX. Књиге које тврде да су о T_EX (укључујући и базичну „*The T_EXBook*”), у суштини говоре о Plain T_EX; и они који верују да раде директно са T_EX у суштини раде са Plain T_EX.

Plain T_EX је оно што се у T_EX терминологији назива *формат*, а састоји се од опсежног скупа макроа, заједно са одређеним правилима синтаксе који одређују како и када треба да се користе. Током времена су уз Plain T_EX развијени и остали *формати*, од којих вреди поменути L^AT_EX, детаљни скуп макроа за T_EX који је 1985. године направио Лесли Лампорт и који представља систем изведен из T_EX који се вероватно најчешће користи у академском, технолошком и математичком свету. ConTeXt је (или је почео да буде), на сличном нивоу као и L^AT_EX, формат настао из T_EX.

Обично уз ове *формате* долази програм који у меморију учитава макрое који их сачињавају пре позивања програма „`tex`” (или машине која се користи за обраду) да обради изворни фајл. Али мада сви ови формати уствари извршавају T_EX, пошто сваки од њих, посматрано из угла корисника, поседује различите инструкције и различита синтаксна правила, сваки од њих можемо схватити као *различити језик*. Сви они узимају инспирацију из T_EX, али се разликују од језика T_EX и једни од других.

1.5 ConTeXt

Мада је у суштини ConTeXt започет као T_EX *формат*, он је данас много више од тога. ConTeXt садржи:

1. Веома опширан скуп T_EX макроа. Ако Plain T_EX садржи око 900 инструкција, ConTeXt их има око 3500; а ако додамо и имена различитих опција које поседују ове команде, говоримо о речнику од око 4000 речи. Речник је тако велики због ConTeXt стратегије да олакшавање његовог учења значи увођење било ког броја синонима за команде и опције.

Ако је потребно постићи неки ефекат, намера је да онда за сваки од начина на који би енглески говорник могао да назове тај ефекат постоји команда или опција која га постиже – што би требало да олакша употребу језика. На пример, ако желите истовремено да добијете црни слог и курзив, ConTeXt вам обезбеђује три инструкције које постижу исти резултат: `\bi`, `\italicbold` и `\bolditalic`.

2. Слично опширан скуп макроа за MetaPost, графички програмски језик изведен из језика MetaFont, језика за дизајн словних ликова који је Кнут развио заједно са програмом T_EX.
3. Разне *скрипте* развијене за језик Perl (најстарије), Ruby (неке такође старе, неке баш и не) и Lua (најсвежије).
4. Интерфејс који интегрише T_EX, MetaPost, Lua и XML, који омогућава писање документа користећи било који ид ових језика, или мешање елемената из неког од њих.

Да ли сте разумели претходно објашњење? Не брините о томе. У њему сам употребио доста компјутерског жаргона и поменуо многе програме и језике. Да бисте користили ConTeXt, није неопходно да познајете све његове разне компоненте. Битна ствар у овој фази учења је да имате на уму да систем ConTeXt интегрише многе алате из различитих извора тако да сви заједно чине *словослагачки систем*.

Ова последња особина интеграције алата различитог порекла је разлог што кажемо да је ConTeXt „хибридна технологија” намењена словослагању докумената. Ја сматрам да то ConTeXt претвара у изузетно напредан и моћан систем.

Мада је ConTeXt много више од колекције T_EX макроа, он је још увек заснован на систему T_EX, па је то разлог што се овај документ, за који не тврдим да је нешто више од *увода*, фокусира на ту чињеницу.

С друге стране, ConTeXt је модернији од T_EX система. Када је креиран T_EX, компјутери су тек почели да се појављују и били смо далеко од тога да схватимо како ће изгледати (постати) свет интернета и мултимедије. У овом погледу ConTeXt природно интегрише неке од ствари које су одувек биле нешто као страно тело у систему T_EX, као што је укључивање графике, управљање бојама, хиперлинкови у електронским документима, подразумевање величине папира која је погодна за документ намењен приказу на екран, итд.

1.5.1 Кратка историја система ConTeXt

ConTeXt је рођен отприлике у 1991. години. Направили су га Ханс Хаген и Тон Отен у холандској компанији за дизајнирање и обраду докумената која се назива „*Pragma Advanced Document Engineering*”, што се обично скраћује на Pragma ADE. Почео је као колекција T_EX макроа који су имали холандска имена и незванично се звао *Pragmatex*, намењен нетехнички образованим запосленима у компанији који су морали да управљају многим детаљима уређивања словослагачких докумената и који нису навикли на употребу језика за означавање или интерфејса на неком језику који није холандски. Због тога је прва верзија система ConTeXt написана на холандском. Идеја је била да се креира довољан број макроа са униформним и конзистентним интерфејсом. *Пакет* је отприлике у 1994. години постао довољно стабилан да се напише корисничко упутство на холандском, па је у 1996., на иницијативу Ханса Хагена, почео да узима име „ConTeXt”. Тврди се да ово име значи „Текст са T_EX” (употребом латинског предлога ‘con’ који значи ‘са’), али је у исто време и игра речи на енглеском (и холандском) од речи „контекст”. Дакле, иза имена стоји трострука игра речи у којој се налазе „T_EX”, „text” и „context”.

Дакле, пошто је његово име засновано на игри речи, ConTeXt би требало да се изговара ‘контекст’ а не ‘контехт’, јер би се на тај начин изгубила игра са речима.

Превод интерфејса на енглески је почео отприлике у 2005. години, па је тако настала верзија позната под именом ConTeXt Mark II, где се ‘II’ објашњава тиме што је у главама програмера претходна верзија на холандском била Верзија ‘I’, мада се званично никада није тако звала. Када се интерфејс превео на енглески, употреба система је почела да се шири ван Холандије, па је интерфејс преведен на и на друге европске језике, као што су француски, немачки, италијански и румунски. Ипак, „званична” документација система ConTeXt је обично базирана на енглеској верзији, и то је верзија којом се бави овај документ.

У својој почетној верзији, ConTeXt Mark II је радио са pdfTeX *TeX* машином. Али касније, појавом XeTeX *TeX* машине, ConTeXt Mark II је измењен тако да се дозволи употреба ове нове машине којом је уведено више предности у односу на pdfTeX. Али када је неколико година касније стигла LuaTeX, донета је одлука да се ConTeXt интерно реконфигурише тако да интегрише све нове могућности које је нудила ова нова машина. И тако је рођен ConTeXt Mark IV, представљен 2007. године, непосредно након представљања LuaTeX машине. Врло је вероватно да је један од одлучујућих фактора за одлуку да се ConTeXt реконфигурише тако да се прилагоди за LuaTeX био тај што су два од три главна програмера система ConTeXt, Ханс Хаген и Тако Хекватер, такође били део главног тима који је развијао LuaTeX. То је разлог што су ConTeXt Mark IV и LuaTeX били рођени у исто време и заједно развијани. Између ConTeXt и LuaTeX постоји синергија која не постоји ни у једном другом деривату система TeX; сумњам да било који други може боље да искористи предности LuaTeX машине од система ConTeXt.

Постоји много разлика између Mark II и Mark IV, мада је већа њих *интерна*, то јест, оне се тичу начина на који макрои функционишу на ниском нивоу, тако да се те разлике из угла корисника ни не примећују: име и параметри макроя остају исти. Међутим, неке разлике које утичу на интерфејс и приморавају да се ствари у једној верзији раде другачије него у другој. Постоји релативно мало таквих разлика, али оне утичу на важне аспекте, као на пример, кодирање улазног фајла, или обрада фонтова инсталираних на систему.



Било би заиста корисно када би негде постојао документ који објашњава (или барем наводи) битне разлике између Mark II и Mark IV. На пример, у ConTeXt викију, за сваку ConTeXt команду постоје *две врсте синтаксе* (које су често идентичне). Претпостављам да једна важи за Mark II а друга за Mark IV; па према овој претпоставци, такође претпостављам да је *прва верзија* за Mark II. Али истина је да нам вики не говори ништа о овоме.

Чињеница да има мало разлика на нивоу језика значи да у већини случајева, уместо да говоримо о две верзије, говоримо о две „варијанте“ система ConTeXt. Без обзира на то да ли их зовете овако или онако, чињеница је да у општем случају документ припремљен за Mark II не може глатко да се компајлира са Mark IV и обрнуто; а ако документ меша обе верзије, највероватније се неће исправно компајлирати ни на једној верзији; па то повлачи да аутор изворног фајла мора донети одлуку да ли жели да пише за Mark II или за Mark IV.

Ако радимо са различитим верзијама система ConTeXt, добар трик за прављење разлике на први поглед између фајлова намењених за Mark II и оних намењених за Mark IV је да се употреби различита екстензија за име фајла. Тако, на пример, свим фајловима које пишем за Mark II, стављам „.mkii“ као екстензију, а „.mkiv“ за оне написане за Mark IV. Истина је да ConTeXt очекује да сви изворни фајлови имају „.tex“ екстензију, али екстензију фајла можемо да променимо ако је експлицитно наведемо када позивамо ConTeXt над фајлом.

ConTeXt дистрибуција инсталирана на викију, „ConTeXt Standalone“, садржи обе верзије, па да би се спречила забуна – претпостављам – користи различиту команду за компајлирање фајла сваке верзије. Mark II компајлира командом „texexec“, а Mark IV командом „context“.

Уствари су обе команде, „context“ и „texexec“, *скрипте* са различитим опцијама које извршавају Lua *скрипту* „mtxrun“.

Данас је Mark II замрзнут, а Mark IV наставља да се развија, што значи да се нове верзије Mark II објављују само када се пронађу грешке или проблеми који морају да се исправе, док се нове верзије Mark IV редовно објављују; понекада два или три пута месечно, мада у већини случајева „нове верзије“ не уносе уочљиве измене језика, већ су ограничене на унапређење имплементације команде на ниском нивоу, или на ажурирање неких од многих упутстава која се испоручују у дистрибуцији. Свеједно, ако инсталирамо развојну верзију – што топло препоручујем, а то је и оно што се подразумевано инсталира са „ConTeXt Standalone“ – има смисла да с времена на време ажурирамо своју верзију (погледајте [Додатак А](#) у којем се говори како ажурирати „ConTeXt Standalone“ верзију).

LMTX и остале алтернативне Mark IV имплементације

Програмери система ConTeXt су природно неуморни, тако да нису обуставили развој система ConTeXt са Mark IV; још увек се тестирају нове верзије и експериментише се са њима, мада се у општем случају оне врло мало разликују од Mark IV и не испољавају некомпатибилност компајлирања која постоји између Mark IV and Mark II.

Тако да је развијено неколико малих варијанти Mark IV под називима Mark VI, Mark IX и Mark XI. Од свих њих, на ConTeXt викију сам успео да пронађем само малу референцу на Mark VI, где се каже да је разлика у односу на Mark IV само у могућности дефинисања команди доделом именованих параметара а не бројева, као у традиционалном систему TeX, дакле на начин како се то обично ради у скоро свим програмским језицима.

Верујем да је у ConTeXt свету појава нове верзије под називом LMTX много важнија од ових малих варијација. LMTX је акроним од luametaTeX: нове TeX машине која представља поједностављену верзију LuaTeX, развијена са циљем уштеде компјутерских ресурса; што значи да LMTX захтева мање меморије и мање процесорске снаге од ConTeXt Mark IV.

LMTX је представљен у пролеће 2019. године и може да се претпостави да неће захтевати било какву спољашњу измену Mark IV језика. За аутора овог документа није било никакве разлике док је радио на њему; али када се компајлира, потребно је да се изабере да ли желите то да радите са LuaTeX, или са luametaTeX. У [додатку А](#), који се тиче инсталације система ConTeXt, приказана је процедура за доделу другачијег имена команде за сваку од инсталација ([одељак 3](#)).

1.5.2 ConTeXt наспрам L^ATeX

Пошто је L^ATeX најпопуларнији формат изведен из TeX, неизбежно је поређење између њега и система ConTeXt. Јасно је да говоримо о различитим језицима који су донекле у вези, јер и један и други проистичу из TeX језика; однос је сличан ономе између, рецимо, шпанског и француског језика; језика који имају заједничко порекло (латински) што значи да су њихове синтаксе *сличне* и да за многе речи у сваком од ових језика постоји врло слична у оном другом. Али за разлику од ове *породичне сличности*, L^ATeX и ConTeXt се разликују у својој филозофији и имплементацији, пошто су почетни циљеви сваког од њих у некој мери супротни. L^ATeX тврди да олакшава употребу TeX, издвајајући аутора од конкретних типографских детаља тако да може да се концентрише на садржај, остављајући систему L^ATeX типографске детаље. Ово значи да се по цену поједностављења употребе система TeX ограничава огромна флексибилност самог TeX тако што се унапред дефинишу основни формати и ограничава број типографских проблема које аутор мора да реши. Супротно од ове филозофије, ConTeXt је рођен у компанији која је посвећена словослагању докумената. Стога, нипошто се не жели изоловање аутора од типографских детаља, већ је циљ да се аутору омогући апсолутна и комплетна контрола над њима. Да би се то постигло, ConTeXt обезбеђује униформни, конзистентан интерфејс који је много ближи оригиналном духу система TeX него што је то L^ATeX.

Ова разлика у филозофији и основним циљевима се затим преноси на разлику у имплементацији. L^ATeX тежи да упрости ствари колико год је то могуће, тако да нема потребе да користи све ресурсе система TeX. На неки начин, његово језгро је прилично једноставно. Па када се јави потреба да се његове могућности прошире, неопходно је да се експлицитно напише *пакет* који то имплементира. Ова *пакетизација* придружена систему L^ATeX је у исто време и врлина и мана: врлина је, јер услед огромне популарности система L^ATeX, уз великодушност његових корисника, то значи да је скоро све што можемо пожелети да урадимо неко пре нас већ имплементирао, па постоји пакет за то; али је такође и мана, јер често ови пакети нису компатибилни међусобно и синтакса коју користе није увек униформна. Ово значи да рад са системом L^ATeX захтева константну претрагу кроз хиљаде постојећих пакета који могу да задовоље неку потребу, уз обезбеђивање да сви заједно раде како се очекује.

За разлику од једноставности L^ATeX језгра, коју допуњава његова проширивост кроз пакете, ConTeXt је дизајниран тако да су у њега укључене све – или скоро све – типографске могућности система TeX, тако да је његова концепција више монолитна, мада у исто време и више модуларна. ConTeXt језгро нам дозвољава да урадимо скоро све, уз гаранцију да неће бити некомпатибилности између различитих алата, неће бити потребно да се истраже проширења за испуњавање неке потребе, а синтакса језика се не мења само из разлога што нам је потребан одређени алат.

Тачно је да ConTeXt поседује нешто што се назива *модули* проширења, за које неко може да каже како обављају функцију сличну L^ATeX пакетима, али у стварности они функционишу на различите начине: ConTeXt модули су дизајнирани тако да само укључе додатне могућности које, из разлога што су још у експерименталној фази, још увек нису постале део језгра, или да се дозволи приступ проширењима које је направио неко ван ConTeXt развојног тима.

Ја лично не сматрам да је било која од ове две *филозофије* боља од оне друге. Питање зависи од корисничког профила и тога шта он или она жели. Ако корисник не жели да се носи са типографским проблемима, већ једноставно да прави стандардизоване документе високог квалитета, вероватно би било боље да се определи за систем као што је L^ATeX; с друге стране, корисник који воли да експериментира, или којем је потребна контрола све до најситнијег детаља документа, или неко ко

је осмислио специјалан распоред за документ, вероватно треба да користи систем као што је ConTeXt, у којем аутор има потпуну контролу; наравно, уз ризик да не зна како исправно да користи ту контролу.

1.5.3 Добро разумевање динамике рада у систему ConTeXt

Када радимо са системом ConTeXt, увек почињемо тако што пишемо текст фајл (који називамо *изворни фајл*), у којем се, заједно са садржајем финалног документа, налазе и инструкције (у ConTeXt-жаргону) које наводе како тачно желимо да се документ форматира: општи изглед његових страница и пасуса, маргине које желимо да доделимо одређеним пасусима, фонт којим желимо да се прикаже текст, фрагменти које желимо да се прикажу употребом другачијег фонта, итд. Када напишемо изворни фајл, у терминалу над њим извршавамо програм „context”, који ће га обрадити и генерисати из њега другачији фајл у којем ће садржај нашег изворног документа бити формиран сагласно са инструкцијама које смо за то навели у изворном фајлу. Овај излазни фајл би могао да се пошаље у (комерцијалну) штампарију, прикаже на екрану, постави на интернет или да се дистрибуира контактима, пријатељима, клијентима, учитељима, ученицима... или другим речима, свима којима је намењен.

Ово значи да када ради у систему ConTeXt аутор ради са фајлом чији изглед нема везе са изгледом финалног документа: фајл са којим аутор непосредно ради је текст фајл који није типографски формиран. Тако да ConTeXt функционише на другачији начин од програма познатих под именом *текст процесори*, који још у време писања приказују коначни изглед документа који се уређује. У почетку ће онима који су навикли на текст процесоре начин рада у систему ConTeXt бити чудан, па ће чак бити потребно и извесно време да се навикну на то. Међутим, када се неко навикне на такав рад, он схвата да је уствари овај другачији начин рада који прави разлику између радног фајла и коначног резултата, из многих разлога у суштини предност. Ја ћу овде, без икаквог одређеног редоследа да истакнем неке од тих предности:

1. Текст фајлови су ‘лакши’ за обраду од бинарних фајлова текст процесора, за уређивање је потребно мање компјутерске меморије, мања је вероватноћа да ће се искварити и не постају нечитљиви када се промени верзија програма који их је креирао. Компатибилни су са било којим оперативним системом и могу да се уређују многим текст едиторима, тако да за рад са њима није потребно да компјутерски систем има инсталиран програм који је креирао фајл: биће довољан било који други програм за уређивање; а сваки компјутерски систем увек поседује неки програм за уређивање текста.
2. Прављење разлике између радног документа и финалног документа помаже да се одвоји стварни садржај документа од онога што одређује његов изглед, па се аутору омогућава да се у фази креирања концентрише на садржај, а да се на изглед фокусира у словослагачкој фази.
3. Омогућава брзу и прецизну измену изгледа документа, пошто се то ради ConTeXt командама које се једноставно проналазе у фајлу.
4. С друге стране, ова могућност измене изгледа дозвољава да се из једног садржаја једноставно креирају две (или више) различите верзије: можда је једна верзија оптимизирана за штампу на папиру, а друга дизајнирана за приказ на екрану, подешена на његову величину и садржи хиперлинкове који немају смисла на папирном документу.
5. Могу лако да се спрече типографске грешке које су честе у текст процесорима, као што је проужавање курзива иза последњег карактера речи.
6. Пошто се радни фајл не дистрибуира и намењен је ‘само за наше очи’, могуће је да се унесу белешке и запажања, коментари, без бојазни да ће се појавити у коначном формираном фајлу који се дистрибуира.

7. Квалитет који може да се постигне истовременом обрадом комплетног документа је много већи од квалитета који се постиже програмом који типографске одлуке мора да донесе док се документ уноси.
8. И тако даље.

Све горе наведено значи, с једне стране, да када се навикнемо на рад у систему ConTeXt, постајемо много ефикаснији и продуктивнији, а с друге стране, да је типографски квалитет који можемо постићи много већи од онога који може да се постигне такозваним *текст процесорима*. И мада је тачно да се они лакше користе, чињеница је да се не користе *много* лакше. Тачно је да систем ConTeXt, као што смо раније поменули, садржи 3500 инструкција, обичан корисник система ConTeXt не мора све да их зна. Да бисмо урадили оно што се обично ради текст процесорима, потребно је да знамо само инструкције које нам омогућавају да назначимо структуру документа, неколико инструкција које се тичу уобичајених типографских ресурса, као што су црни слог и курзив, и можда како да генеришемо листу, или фусноту. Све у свему, не више од 15 или 20 инструкција ће нам омогућити да урадимо скоро све ствари које се раде текст процесором. Остале инструкције нам омогућавају да урадимо ствари које обично не можемо да урадимо текст процесором, или које су компликоване за извођење. Можемо рећи да иако је теже научити коришћење система ConTeXt него коришћење текст процесора, то је зато што системом ConTeXt можемо да урадимо много више.

1.5.4 Добијање помоћи у вези система ConTeXt

Док смо још увек почетници, [вики](#) је несумњиво најбоље место да се добије помоћ у вези система ConTeXt. Он је пун примера и има одличан систем претраге, посебно ако добро разумете енглески језик. Помоћ такође можемо да пронађемо и на интернету, али овде ћемо имати проблем са игром речи у имену ConTeXt, јер ће претрага на реч „context” да врати милионе резултата од којих већина нема никакве везе са оним што нас интересује. Да бисте пронашли информације о систему ConTeXt морате да додате нешто уз реч „context”; на пример, „tex”, или „Mark IV” или „Hans Hagen” (један од креатора система ConTeXt) или „Pragma ADE”, или нешто слично. Такође би могло бити од користи да информације тражите употребом имена викија: „contextgarden”.

Када научимо нешто више о систему ConTeXt, можемо да консултујемо неке од многих докумената који су део „ConTeXt Standalone”, или да чак потражимо помоћ на [TeX - LaTeX Stack Exchange](#), или на мејлинг листи за ConTeXt ([NTG-context](#)). Ову листу прате људи који знају већину ствари у вези система ConTeXt, али правила добре сајбер етикеције захтевају да смо се пре постављања питања и сами заиста потрудили да дођемо до одговора.

Глава 2

Наш први изворни фајл

Садржај: 2.1 Припрема експеримента: неопходни алати; 2.2 Сам експеримент; 2.3 Структура фајла нашег примера; 2.4 Неки додатни детаљи у вези покретања команде „context”; 2.5 Управљање грешкама;

Ово поглавље је посвећено нашем првом експерименту и објасниће основну структуру ConTeXt документа, као и најбоље стратегије за исправљање потенцијалних грешака.

2.1 Припрема експеримента: неопходни алати

Да бисмо написали и компајлирали први изворни фајл, потребно је да на систему имамо инсталиране следеће алате.

1. **Текст едитор** за писање тест фајла. Постоји много текст едитора и тешко је замислити оперативни систем који већ нема неки преинсталиран. Можемо користити било који од њих: постоје једноставни, сложенији, моћнији, неки које морате да платите, неки бесплатни (тј. *gratis*), неки слободни (као *libre*), неки који су специјализовани за T_EX системе, а неки служе за општу примену, итд. Ако смо навикли да радимо у одређеном едитору, најбоље би било да наставимо да га користимо; ако у овом тренутку нисте навикли да радите ни са једним, мој савет је да најпре пронађете једноставни едитор, тако да уз комплексност учења система ConTeXt не морате учити и како да користите текст едитор. Мада је тачно да су често програми који су најкомпликованији за учење они који су најмоћнији.

Овај текст сам написао користећи GNU Emacs, један од најмоћнијих и свестраних едитора опште намене који постоји; тачно је да има одређене специфичности и особине које га одвраћају од потенцијалних корисника, али у суштини има више „Emacs-љубитеља” него „Emacs-хејтера”. Постоји GNU Emacs екстензија под називом AucTeX која служи за рад са T_EX фајловима или неким од његових деривата. Она едитору обезбеђује неке врло интересантне додатне алате, мада је AucTeX у суштини боље припремљен за рад са L^AT_EX него са ConTeXt фајловима. GNU Emacs у комбинацији са AucTeX би могао да буде одлична опција ако не знате који едитор да изаберете; и један и други су *libre* програми, тако да постоје верзије са све оперативне системе. Уствари, изјављивање како је GNU Emacs *софтвер libre* је потцењивање, јер овај програм боље него било који други отеловљује дух и значење *слободног софтвера*. На крају, његов главни програмер је био Ричард Столмен оснивач и идеолог GNU пројекта и *Фондације слободног софтвера*.

Слично као и GNU Emacs + AucTeX, остале добре опције, у случају да не знате шта да изаберете, су *Scite* и *TexWorks*. Овај први, мада је едитор опште намене који није посебно дизајниран за рад са ConTeXt фајловима, једноставно се прилагођава и, како је то едитор који у општем случају користе ConTeXt програмери, „ConTeXt Standalone” садржи конфигурационе фајлове за овај едитор које је написао сам Ханс Хаген. С друге стране, *TexWorks* је брз едитор који је специјализован за обраду T_EX фајлова и фајлова језика изведених из њега. Веома је једноставно да се подеси за рад са ConTeXt и „ConTeXt Standalone” такође доставља његову конфигурацију.

Који год едитор да изаберемо, једина ствар коју не смемо да користимо као текст едитор је *текст процесор* као што је, на пример, OpenOffice Writer или Microsoft Word. Ови програми, по мом

мишљењу преспори и тешки, могу, ако се то експлицитно назначи, да сачувају фајл као 'чисти текст (txt)', али они нису дизајнирани за тако нешто, па ћемо највероватније завршити са фајлом сачуваним у неком бинарном формату који није компатибилан са системом ConTeXt.

2. **ConTeXt** дистрибуција за обраду нашег тест фајла. Ако на вашем систему већ постоји T_EX (или L^AT_EX) инсталација, врло је вероватно да је инсталирана и верзија система ConTeXt. Да бисте то проверили, довољно је да покренете терминал и откуцате

```
$> context --version
```

НАПОМЕНА за оне који нису раније користили терминале, прва два карактера која сам написао („\$>“) се не пишу у терминал. Једноставно сам приказао оно што се назива терминалски одзив; мали трепћући знак који означава да терминал очекује ваше инструкције.

Ако је верзија ConTeXt система већ инсталирана, појавиће се нешто слично следећем:

```
mtx-context | ConTeXt Process Management 1.03
mtx-context |
mtx-context | main context file: /home/jq/context/LMTX/tex/texmf-context/
            | tex/context/base/mkiv/context.mkiv
mtx-context | current version: 2020.04.30 11:15
mtx-context | main context file: /home/jq/context/LMTX/tex/texmf-context/
            | tex/context/base/mkiv/context.mxl
mtx-context | current version: 2020.04.30 11:15
```

Последња линија нас обавештава о датуму објављивања инсталиране верзије. Ако је то сувише старо, требало би или да је ажурирамо, или да инсталирамо нову верзију. Препоручујем да се инсталира дистрибуција под називом „ConTeXt Standalone“ чија упутства за инсталацију можете да пронађете на [ConTeXt викију](#). Преглед свега овога можете пронаћи у [додатку А](#).

3. **Читач PDF фајлова**, тако да на екрану можемо да видимо резултат нашег експеримента. На Windows и Mac OS увек постоји Adobe Acrobat Reader. Ако подразумевано није инсталиран (или није када сам престао да користим Microsoft Windows пре више од 15 година), али то уради први пут када покушате да отворите PDF фајл, тако да је највероватније преинсталиран. Linux/Unix системи немају тренутну верзију Acrobat Reader програма, али вам није ни потребна јер је доступно буквално на десетине бесплатних и веома добрих PDF читача. Осим тога, скоро увек је неки од њих подразумевано инсталиран на овим системима. Из разлога брзине и једноставности употребе, мој омиљени је MuPDF; мада има неке лоше стране ако користите друге језике осим енглеског са акцентованим карактерима, и не дозвољава вам да изаберете текст или да документ пошаљете на штампач; то је једноставно читач; али веома брз и лак за употребу. Када су ми потребне могућности које не MuPDF не поседује, обично користим или Okular, или qPdfView. Али још једном, то је ствар укуса: можете изабрати шта год вам одговара.

Можемо да изаберемо свој едитор, свој PDF читач, своју ConTeXt дистрибуцију... Добродошли у свет слободног *libre софтвера*!

2.2 Сам експеримент

Писање изворног фајла

Ако су вам алати поменути изнад већ доступни, потребно је да отворите свој текст едитор и да њиме креирате фајл који ћемо назвати „odmor.tex“. Као садржај фајла ћемо написати ово што следи:


```
% Прва линија документа

\mainlanguage[sr] % Језик = српски

\setuppapersize[S5] % величина папира

\setupbodyfont
[dejavu,12pt] % Фонт = DejaVu Serif, 12 тачака

\setuphead      % Формат поглавља
[chapter]
[style=\bfc]

\starttext % Почетак садржаја документа

\startchapter
[title=Годишњи одмор...]

Пепсико, Пепсико, хајдемо у Мексико.
Никада, никада, јер ја немам долара.
Ко има долара, купа се на плажи.
Ко нема долара, кући на гаражи!
Пепсико, Пепсико, хајдемо у Мексико.

\stopchapter

\stoptext % Крај документа
```

Док се пише, није битно ако се било шта промени, посебно ако се додаје празни простор или преломи линија. Оно што је битно је да се речи које следе након „\” напишу тачно како је приказано, као и садржај унутар витичастих заграда. Остатак може бити другачији.

Кодирање карактера фајла

Када напишемо ово изнад, фајл треба да сачувамо на диск, и обезбедимо да је кодирање карактера UTF-8. Данас је ово кодирање стандардно. У сваком случају, ако нисмо сигурни, сам едитор нам може показати кодирање, па ако је потребно, можемо и да га променимо. Начин како се то ради, очигледно зависи од самог едитора који се користи. На пример, у GNU Emacs се то ради притиском комбинације тастера CTRL-X, па затим Ентер након кога следи ‘f’, у последњој линији прозора (коју GNU Emacs назива *мини-бафер*) ће се појавити порука која од нас захтева ново кодирање и приказује нам које је текуће кодирање. У осталим едиторима се кодирање обично налази у „Save as” менију.

Када смо проверили да је кодирање исправно и сачували фајл на диск, затварамо едитор и фокусирамо се на анализирање онога што смо написали.

Поглед у садржај нашег првог изворног фајла написаног за ConTeXt

Прва линија почиње карактером „%”. Ово је резервисани карактер који систему ConTeXt говори да не обрађује текст који се налази између тог карактера и краја линије у којој се он налази. Ово је корисно када желимо да напишемо коментар на изворни фајл који само аутор може да чита, пошто он не постаје део финалног документа. У овом примеру сам употребио тај карактер да привучем пажњу на одређене линије, објашњавајући шта оне раде.

Линије које почињу са карактером „\”, још једним од резервисаних карактера у систему ConTeXt који означава да оно што следи представља име команде. Овај пример приказује низ команди које се налазе у ConTeXt изворном фајлу: језик на којем је написан документ, величина папира, фонт који ће се користити за приказ документа и начин на који ће се форматирати поглавља. Касније, у наредним поглављима ћемо видети детаље ових команди, али за сада је битно само да читалац види како изгледају ове команде: оне почињу са „\”, затим долази име команде, па онда између витичастих заграда или великих заграда, у зависности од ситуације, подаци који су команди потребни да

произведе жељени ефекат. Између имена команде и великих или витичастих заграда може да стоји празан простор или преломи линија.

У деветој линији нашег примера (бројим само линије које у себи садрже неки текст) налази се важна `\starttext` команда: она систему ConTeXt говори да од те тачке почиње садржај документа; а у последњој линији нашег примера, видимо команду `\stoptext` која говори да је ово место ка којем се документ завршава. Ово су две веома важне команде о којима ћу ускоро имати још по нешто да кажем. Између њих се налази садржај документа, који у овом случају представља шаљиву дечију песмицу. Написао сам је у форми прозе да бисмо видели начин на који систем ConTeXt форматира пасус.

Обрада изворног фајла

За наредни корак, након што смо потврдили да је систем ConTeXt исправно инсталиран на нашем систему, морамо да отворимо терминал у оном директоријуму у који је сачуван фајл „odmor.tex”.

Многи текст едитори нам дозвољавају да компајлирамо документ на којем смо радили без потребе да отварамо терминал. Ипак, *канонска* процедура за обраду документа системом ConTeXt наводи да се то ради из терминала, директним извршавањем програма. Кроз цео овај документ ћу то радити на овај начин (или претпоставити да је урађено тако) из различитих разлога; први је што не могу да знам који едитор користи читалац. Али најважнији је онај што употреба терминала омогућава приступ екранском излазу програма „context” и могуће је да се виде поруке које исписује програм.

Ако је ConTeXt дистрибуција коју смо инсталирали „ConTeXt Standalone”, пре било чега морамо да извршимо *скрипту* која терминалу говори путању и локацију фајлова који су потребни да се ConTeXt извршава. На Linux/Unix системима, то се ради тако што се изврши следећа команда:

```
$> source ~/context/tex/setuptex
```

под претпоставком да смо ConTeXt инсталирали у директоријум под именом „context”.

Безано за извршавање *скрипте* о којем смо управо говорили, погледајте шта у [додатку А](#) пише о инсталацији „ConTeXt Standalone”.

Када се у меморију учитају променљиве потребне да се покрене „context”, можемо да га извршимо. То радимо тако што у терминалу откуцамо

```
$> context odmor
```

Без обзира на то што се изворни фајл назива „odmor.tex”, приметите да смо приликом позивања „context” изоставили екстензију фајла. У случају да смо фајл назвали „odmor.mkiv”, на пример (што обично и радим како бих могао да знам да је тај фајл написан за Mark IV), морали бисмо експлицитно да наведемо екстензију, наводећи „context odmor.mkiv”.

Када у терминалу покренемо „context”, на екрану ће се појавити неколико десетина линија које нам говоре шта ConTeXt ради. Ове информације се појављују брзином коју човек не може да испрати, али то не треба да вас забрине, јер се исте информације чувају и у помоћном фајлу са екстензијом „.log”. Он се генерише у време обраде, па ако је неопходно, касније можемо на миру да га погледамо.

Неколико секунди касније, ако смо изворни фајл написали без неке озбиљне грешке, терминалске поруке ће се завршити. Последња порука нас обавештава колико је трајало компајлирање фајла. Први пут је за компајлирање потребно мало више времена, јер ConTeXt мора да учита у меморију одређене фајлове који му говоре који фонтови се користе, док су за накнадна компајлирања они већ учитани. Када се појави последња порука која обавештава о дужини компајлирања, обрада је завршена. Ако је све прошло како треба, сада ће се у директоријуму у којем смо извршили „context” појавити три додатна фајла:

- odmor.pdf
- odmor.log
- odmor.tuc

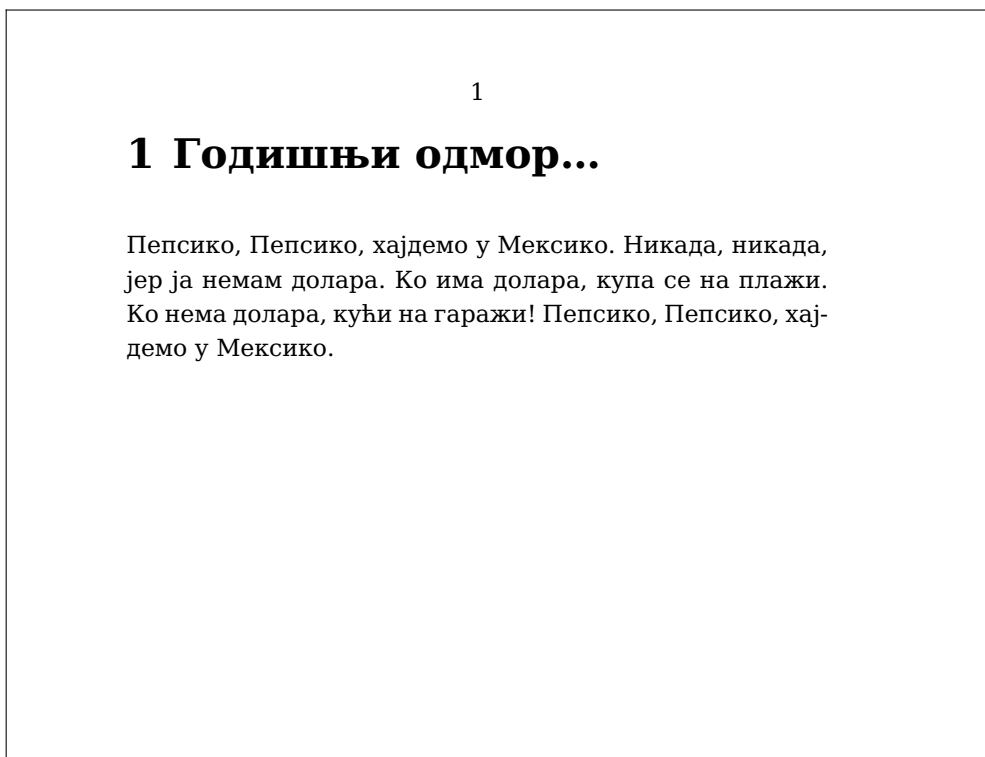
Први од ова три је резултат обраде, или другим речима, коначни форматирани PDF. Други је „.log” фајл у којем се налазе све информације приказане на екран током обраде изворног фајла; трећи је помоћни фајл који ConTeXt генерише током компајлирања и који користи за креирање индекса и унакрсних референци. За сада, ако је све прошло како треба, можемо да обришемо и (odmor.log и odmor.tuc). Ако је било проблема, садржај ових фајлова ће нам помоћи да пронађемо где је узрок и како да дођемо до решења.

Ако нисмо добили ове резултате, узрок је вероватно јер:

- или нисмо исправно инсталирали ConTeXt дистрибуцију, па том случају, када у терминалу покушамо да извршимо команду „context”, видимо поруку „command unknown”.
- или наш фајл није UTF-8 кодиран, па долази до грешке у обради.
- или је можда ConTeXt који је инсталиран на систему Mark II. У овој верзији не можемо да користимо UTF-8 кодирање ако се то експлицитно не наведе у изворном фајлу. Могли бисмо да направимо изворни фајл тако да се исправно компајлира, али пошто се овај увод односи на Mark IV, нема смисла да се настави рад са Mark II: најбоље би било да се инсталира „ConTeXt Standalone”.
- или смо направили грешку у изворном фајлу док смо писали име команде или уносили податке потребне команди.

Ако након почетка извршавања „context” команде у терминалу крену да се емитију поруке, па се зауставе, а не појави се одзив, пре него што наставимо морамо да притиснемо CTRL-X да би прекинули извршавање ConTeXt које је зауставила грешка.

Затим морамо открити шта се догодило и решимо проблем, па наставимо то да радимо све док се компајлирање не обави успешно.



Слика 2.1 Годишњи одмор...

На слици 2.1 видимо садржај фајла „odmor.pdf”. Такође можемо видети да је ConTeXt нумерисао страницу и поглавље, написао текст фонтом који смо тражили. ConTeXt ће подразумевано да прелама речи на крају реда сагласно правилима изабраног језика, а у нашем случају прва линија наводи (`\mainlanguage[sr]`).

Да сумирамо: ConTeXt је трансформисао изворни фајл и генерисао фајл у којем имамо документ форматиран сагласно инструкцијама у изворном фајлу. Нестали су сви евентуални коментари, и што се тиче команди, оно што сада имамо није име команде, већ резултат извршавања команде.

2.3 Структура фајла нашег примера

У пројекту који се развија само у једном изворном фајлу, структура је веома једноставна и означена командама `\starttext ... \stoptext`. Све што се налази између прве линије фајла и команде `\starttext` се назива *преамбула*. Стварни садржај документа се умеће између команди `\starttext` и `\stoptext`. У нашем примеру се преамбула састоји из три глобалне конфигурационе команде: једна наводи језик нашег документа, (`\mainlanguage`), друга наводи величину страница (`\setuppapersize`) која је у нашем случају „S5” и представља димензије компјутерског екрана, а трећа команда (`\setuphead`) омогућава да конфигуришемо изглед наслова поглавља.

Тело документа се налази између команди `\starttext` и `\stoptext`. Ове команде наводе почетну и крајњу тачку текста који треба да се обради: између њих треба да поставимо сав текст који желимо да ConTeXt обради, уз команде које не треба да утичу на комплетан документ, већ само на одређене делове. За сада, претпоставимо да су `\starttext` и `\stoptext` команде обавезне у сваком ConTeXt документу, мада ћемо касније, када будемо говорили о пројектима у више фајлова (одељак 4.6), видети да постоје неки изузеци.

2.4 Неки додатни детаљи у вези покретања команде „context”

Команда „context” којом смо раније почели обраду нашег првог изворног фајла је уствари Lua скрипта, што значи да је то мали Lua програм који после обављања одређених провера позива LuaTeX, јер он обрађује изворни фајл.

Команду „context” можемо да позовемо уз разне опције. Оне се наводе непосредно након имена команде, тако што се откуцају две цртице. Ако желимо да наведемо више од једне опције, раздвајамо их размаком. Опција „help” исписује листу свих опција уз кратко објашњење сваке од њих:

```
$>context --help
```

Следе неке од интересантнијих опција:

interface: Као што сам већ напоменуо у уводном поглављу, ConTeXt интерфејс је преведен на разне језике. Подразумевани интерфејс је енглески, међутим, ова опција нам омогућава да користимо холандски (nl), француски (fr), италијански (it), немачки (de) или румунски (ro).

purge, purgeall: Брише помоћне фајлове који су генерисани током обраде.

result=Name: Наводи име које би требало да добије произведени PDF фајл. Подразумевано ће бити исто као и име фајла који се обрађује, са екстензијом .PDF.

usemodule=list: Учитава наведене модуле пре него што покрене ConTeXt (модул је проширење система ConTeXt које није део његовог језгра и које обезбеђује неку додатну могућност).

useenvironment=list: Учитава фајлове окружења пре него што покрене ConTeXt (фајл окружења је фајл са конфигурационим инструкцијама).

version: Приказује верзију система ConTeXt.

help: приказује помоћ у вези опција програма.

noconsole: Спречава слање порука на екран током компајлирања. Међутим, ове поруке се ипак чувају у `.log` фајл.

nonstopmode: Извршава компајлирање без заустављања у случају грешака. Ово не значи да се грешка не генерише, већ да кад ConTeXt наиђе на грешку, чак и неку коју може да опорави, он наставља са компајлирањем све док не стигне до краја, или док не наиђе на грешку од које не може да се опорави.

batchmode: Комбинација претходне две опције. Извршава се без прекида и спречава испис порука на екран.

Сматрам да у првим корацима учења система ConTeXt није добра идеја да се користе последње три опције, јер када дође до грешке нећемо знати где се она налази или шта ју је изазвало. А верујте ми драги читаоци, пре или касније ћете имати грешке током обраде.

2.5 Управљање грешкама

Током рада у систему ConTeXt неизбежно је да се пре или касније појаве грешке током обраде. Грешке можемо да поделимо у следеће четири категорије:

1. **Грешке у писању.** Дешавају се када направимо грешку у куцању имена команде. У овом случају компајлеру шаљемо наредбу коју не разуме. На пример, када уместо да напишемо команду `\TeX` напишемо `\Tex` тако да је последње слово мало 'x', а како ConTeXt прави разлику између малих и великих слова, он види „TeX” и „Tex” као различите речи; или ако се опције функционисања команде поставе у велике заграде уместо у витичасте заграде, или ако покушамо да употребимо резервисане карактере као да су обични итд.
2. **Грешке изостављања.** ConTeXt има инструкције за почетак задатка чији крај морамо такође експлицитно да означимо; као што је резервисани карактер `$` за укључивање математичког режима који траје све док се не искључи, па заборавимо да га искључимо. Онда се генерише грешка када се наиђе на текст или инструкцију која нема смисла у математичком режиму. Исто је и ако започнемо блок текста резервисаним карактером `'{'` или командом `\startHesho` па се касније не пронађе експлицитна команда затварања `'}'` или `\stopHesho`.
3. **Грешке концепције.** Тако зовем грешке које се јаве када се позове команда којој су потребни одређени аргументи, али јој се не доставе, или када није исправна синтакса позива команде.
4. **Грешке ситуације.** Постоје команде које су дизајниране тако да раде само у одређеним контекстима или окружењима и уопште се не препознају ван њих. Ово се посебно дешава у математичком режиму: неке ConTeXt команде раде само док се пишу математичке команде и ако се позову из неког другог окружења, генерисаће грешку.

Шта да радимо када нас „context”, док обрађује, упозори да је дошло до грешке? Очигледно је прва ствар да одредимо шта је грешка. За то је потребно да анализирамо „.log” фајл који је био генерисан за време обраде; мада понекад то није неопходно, јер је грешка такве природе да је тренутно обуставила обраду, па је у том случају порука о грешки видљива у истом терминалу у којем смо и покренули „context”.

На пример, ако смо у нашем тест фајлу „odmor.tex”, грешком уместо `\starttext` написали `\startext` (са једним 't'), што је врло честа грешка, током извршавања команде „context rain” обрада ће да се прекине и у терминалу ћемо видети информације приказане на [слици 2.2](#). Ту можемо видети да су линије нашег изворног фајла нумерисане и да је у једној од њих, у овом случају линији број 14, између броја и линије текста, компајлер додао „>>” да би означио да је у тој линији наишао на грешку. Фајл „odmor.log” ће нам пружити још трагова. У нашем примеру фајл није тако велики, јер се извор који се компајлира знатно скраћен; али у осталим случајевима може да садржи огромну количину информација. Али морамо да уронимо у њих. Ако текст едитором отворимо „odmor.log” видећемо да

```

4
5 \setpapersize[55] % величина папира
6
7 \setbodyfont
8 [dejavu,12pt] % Фонт = DejaVu, 12 тачака
9
10 \setuphead % Формат поглавља
11 [chapter]
12 [style=\bfc]
13
14 >> \starttext % Почетак садржаја документа
15
16 \startchapter
17 [title=Годишњи одмор...]
18
19 Пепсико, Пепсико, хајдемо у Мексико.
20 Никада, никада, јер ја немам долара.
21 Ко има долара, купа се на плажи.
22 Ко нема долара, кући на гаражи!
23 Пепсико, Пепсико, хајдемо у Мексико.
24
mtx-context | fatal error: return code: 256

```

Слика 2.2 Излаз екрана у случају грешке при компајлирању

се у њему налази све што ConTeXt ради. Морамо да пронађемо линију у којој почиње упозорењем о грешки, а за то можемо употребити функцију претраге едитора. Потражићемо „tex error“, па ће нас то довести до следећих линија:

```

tex error      > tex error on line 14 in file |
               ./odmor.tex: Undefined control sequence \undefined

```

```

<line 3.14>
\starttext
% Почетак садржаја документа

```

Напомена: прва линија која нам говори о грешки у фајлу „odmor.log“ је доста дугачка. Да би лепше изгледала, имајући у виду ширину странице, поделио сам је на две линије. Карактер ‘|’ приказује тачку поделе.

Ако обратимо пажњу на три линије поруке о грешки, видећемо да нам прво говоре у којој линији се налази грешка (линија 14) и које је врсте: „Undefined control sequence“, или, што је иста ствар: непознати контролни низ, то јест непозната команда. Наредне линије лог фајла приказују линију 14, подељену на тачки која је проузроковала грешку. Тако да нема сумње да се грешка налази у `\starttext`. Читамо пажљиво и уз мало среће и искуства, схватамо да смо написали „starttext“ уместо „starttext“ (са два ‘t’).

Имајте на уму да су компјутери веома успешни и врло брзи у извршавању инструкција, али веома споро читају наше мисли. А реч „starttext“ није иста као „starttext“. Програм зна како да изврши ову другу, али не и прву. Не зна шта да уради са њом.

У другим случајевима грешка неће моћи тако лако да се открије. Посебно када је грешка у томе да је нешто започето, а није наведено место на којем треба да се заврши. Понекада би, уместо да у „.log“ фајлу тражимо „tex error“, требало да тражимо звездицу. Овај карактер на почетку линије не означава фаталну грешку, већ пре упозорење. Међутим, упозорења могу бити од помоћи код проналажења грешака.

А ако информације у „.log“ фајлу нису довољне, требало би да прођемо кроз главни фајл, део по део, и тражимо грешку. Добра стратегија за то је да се промени локација команде `\stoptext`. Упамтите да ConTeXt прекида обраду текста када наиђе на ову команду. Дакле, ако поставим команду `\stoptext` мање више на половину фајла, па га онда компајлирам, компајлираће се само прва половина фајла; ако се поново јави грешка, онда знам да се она налази у првој половини изворног фајла, а ко се не јави, онда то значи да је у другој половини... па тако, мало по мало, изменом локације команде `\stoptext`, бићемо у могућности да пронађемо место на коме се налази грешка. Када је пронађемо, можемо покушати да је разумемо и исправимо, или ако не разумемо зашто долази до грешке, онда

пошто знамо где долази до грешке, можемо покушати да тај део напишемо на другачији начин и спречимо појаву грешке. Ово друго решење може да се примени само ако смо ми аутор. Ако једноставно радимо словослагање нечијег текста, не можемо да га изменимо и морамо наставити да истражујемо све док не откријемо узроке грешке и могући начин да је исправимо.

У пракси, када се системом ConTeXt прави релативно дугачак документ, он се обично повремено компајлира док се уноси, тако да када дође до грешке, биће мање више јасно да се она јавља у новом делу, јер се последњи пут документ успешно компајлирао. Такође ће бити јасније и зашто долази до грешке.

Глава 3

Команде и остали фундаментални концепти система ConTeXt

Садржај: 3.1 ConTeXt резервисани карактери; 3.2 Саме команде; 3.3 Опсег важења команди; 3.3.1 Командне којима треба или не треба назначити опсег важења; 3.3.2 Команде којима је потребно изричито навођење где почињу и где се завршавају (окружења); 3.4 Опције рада команде; 3.4.1 Команде које могу да раде на неколико различитих начина; 3.4.2 Команде које конфигуришу начин на који раде друге команде (\setupNesto); 3.4.3 Подешавање прилагођених верзија подесивих команди (\defineNesto); 3.5 Резиме синтаксе команде и опција, употреба великих и витичастих заграда приликом позивања; 3.6 Званична листа ConTeXt команди; 3.7 Дефинисање нових команди; 3.7.1 Општи механизам за дефинисање нових команди; 3.7.2 Креирање нових окружења; 3.8 Остали основни концепти; 3.8.1 Групе; 3.8.2 Димензије; 3.9 Метода за самостално учење система ConTeXt;

Већ смо видели да се у изворном фајлу, као и у самом садржају нашег будућег форматираног документа, налазе инструкције које систему ConTeXt објашњавају како желимо да се наш рукопис трансформише. Ове инструкције можемо да назовемо „команде“, „макрои“ или „контролни низови“.

Из угла интерног функционисања система ConTeXt (то јест функционисања система TeX), постоји разлика између *примитива* и *макроа*. Примитива је једноставна инструкција која не може да се разбије у друге једноставније инструкције. Макро је инструкција која се разбија у друге једноставније инструкције, које можда такође могу да се разбију у друге једноставније и тако редом. Већина ConTeXt инструкција су, у ствари, макрои. Из перспективе програмера, разлика између макроа и примитива је важна. Али из перспективе корисника ствар није тако битна: у оба случаја имамо инструкције које се извршавају без потребе да водимо рачуна о начину на који оне функционишу на ниском нивоу. Стога, ConTeXt документација обично говори о *командама* када посматра из угла корисника, а *макроу* када говори из угла програмера. Пошто у овом уводу говоримо само из угла корисника, ја ћу користити било који од ова два израза и посматраћу их као да су синоними.

Команде су налози којима се систему ConTeXt говори да нешто уради; њима ми *контролишемо* ефекат програма. Тако да кнут, отац система TeX, користи израз *контролни низови* када мисли и на примитиве и на макрое, и ја мислим да је то најпрецизнији од свих израза. Користићу га када верујем да је важно направити разлику између *контролних симбола* и *контролних речи*.

ConTeXt инструкције могу у основи да буду или резервисани карактери, или команде у ужем смислу.

3.1 ConTeXt резервисани карактери

Када ConTeXt чита изворни фајл састављен само од текст карактера, а пошто је то чист текст, потребно је на неки начин разликовати текст који треба да се форматира од инструкција које се извршавају над њим. ConTeXt резервисани карактери омогућавају то разликовање. У принципу, ConTeXt ће претпоставити да сваки карактер у фајлу треба да се обради, осим ако није један од 11 резервисаних карактера који се треба да се третирају као *инструкција*.

Само 11 инструкција? Не. Постоји само 11 резервисаних карактера; али пошто један од њих, карактер „\“, има функцију конвертовања карактера или више њих који му непосредно следе у инструкцију, онда је у суштини потенцијални број команди практично неограничен. ConTeXt поседује око

3000 команди (кад се саберу команде које постоје само у Mark II, Mark IV и оне заједничке за обе верзије).

Ово су резервисани карактери:

\ % { } # ~ | \$ _ ^ &

ConTeXt их интерпретира на следећи начин:

- \ Ово је за нас најзначајнији карактер: он означава да све што следи непосредно након њега не сме да се интерпретира као текст већ као инструкција. Он се назива „Означавајући (Escape) карактер” или „Означавајући низ” (мада нема никакве везе са „Esc” тастером који се налази на већини тастатура).¹
- % Говори систему ConTeXt да је све оно што следи до краја линије коментар који не сме да се обрађује или укључи у финални форматирано фајл. Уметање коментара у изворни фајл је изузетно корисно. Коментар може помоћи да се објасни зашто је нешто урађено на одређени начин, а то је заиста корисно у завршеним изворним фајловима у смислу каснијих ревизија, када понекад не можемо да се сетимо зашто смо нешто урадили онако како смо урадили; или такође може да послужи као подсетник у вези нечега што би можда требало да прерадимо. Може чак да се употреби као помоћ за лоцирање одређене грешке у изворном фајлу, јер се постављањем знака коментара на почетак линије она не компајлира, па можемо видети да ли је та линија била узрок грешке; такође може да се користи за чување две различите верзије истог макроа, па да на тај начин добијемо различите резултате након компајлирања; или да спречи компајлирање фрагмента у вези кога нисмо сигурни, а да га не обришемо из изворног фајла, у случају да касније желимо да се вратимо на њега... итд. Једном кад добијемо могућност да изворни фајл садржи текст који само ми видимо и нико други, употреба овог карактера је ограничена само нашом сопственом маштом. Признајем да је ово једна од могућности која ми највише недостаје онда када је једини начин за писање текста употреба текст процесора.
- { Овај карактер отвара групу. Групе су блокови текста на које утичу одређене могућности. О њима ћемо говорити у [одељку 3.8.1](#).
- } Овај карактер затвара групу претходно отворену са {.
- # Овај карактер се користи за дефиницију макроа. Он указује на аргументе макроа. Погледајте [одељак 3.7.1](#) у овом поглављу.
- ~ Уноси размак у документ који спречава прелом линије, што значи да ће две речи раздвојене карактером ~ увек бити у истој линији. О овој инструкцији и местима на којима би требало да се употреби ћемо говорити у [одељку 11.3.1](#).
- | Овај карактер се користи за означавање да две речи спојене раздвајајућим елементом праве сложену реч која може да се подели на слоге у прву компоненту, али не у другу компоненту. Погледајте [одељак 10.4](#).
- \$ Овај карактер је прекидач за математички режим. Он га укључује ако је био искључен, или га искључује ако је био укључен. Када се налази у математичком режиму, ConTeXt примењује неке фонтове и правила која се разликују од уобичајених, чији је циљ оптимизација писања

¹ У компјутерској терминологији се тастер који утиче на интерпретацију наредног карактера назива *escape* (означавајући) карактер. За разлику од тога, *escape тастер* се на тастатурама тако назива јер генерише карактер 27 у ASCII коду, који се у овом кодирању употребљава као *escape* карактер. Данас се употреба *Escape* тастера повезује са идејом отказивања текуће операције.

математичких формула. Мада је писање математике веома битна употреба система ConT_EXt, ја о томе нећу детаљно говорити у овом уводу. Пошто сам писац, није ми до тога!

- Овај карактер се користи да у математичком режиму означи да је оно што следи исписано у индексу. На пример, да бисмо добили x_1 , морамо да напишемо `x_1`.
- ^ Овај карактер се користи да у математичком режиму означи да је оно што следи исписано у експоненту. На пример, да бисмо добили $(x + i)^{n^3}$ потребно је да напишемо `$(x+i)^{n^3}$`.
- & У ConT_EXt документацији пише да је ово резервисан карактер, али не каже зашто. У Plain T_EX овај карактер има у суштини две употребе: користи се да поравна колоне у окружењима основне табеле, а у математичком контексту да оно што следи треба да се третира као обичан текст. У уводном упутству „ConT_EXt Mark IV, an Excursion”, мада не пише за шта служи, постоје примери његове употребе у математичким формулама, додуше не онако како се користи у Plain T_EX, већ да поравна колоне унутар комплексних функција. Пошто сам ја писац, мислим да не могу обавити додатна тестирања којима могу сазнати за шта се тачно користи овај резервисани карактер.



Може се претпоставити да се у избору карактера који би требало да буду резервисани водило тиме да то буду карактери доступни на већини тастатура, али они који се обично не користе у писаном језику. Међутим, мада није тако уобичајено, увек постоји могућност да се неки од њих појави у нашим документима, као на пример, када желимо да напишемо да нешто кошта 100 долара (\$100), или да је у Шпанији 2018. године проценат возача преко 65 година старости био 16%. У овим случајевима не смемо директно да пишемо резервисани карактер, већ морамо да користимо команду која ће исправно исписати резервисани карактер у финални документ. Команда за сваки резервисани карактер може да се пронађе у [табели 3.1](#).

| Резервисани карактер | Команда која га генерише |
|----------------------|---------------------------|
| \ | <code>\backslash</code> |
| % | <code>\%</code> |
| { | <code>\{</code> |
| } | <code>\}</code> |
| # | <code>\#</code> |
| ~ | <code>\lettertilde</code> |
| | <code>\ </code> |
| \$ | <code>\\$</code> |
| _ | <code>_</code> |
| ^ | <code>\letterhat</code> |
| & | <code>\&</code> |

Табела 3.1 Писање резервисаних карактера

Још један начин за добијање резервисаног карактера је командом `\type`. Ова команда шаље свој аргумент у финални документ без било какве обраде, дакле и без интерпретирања. Текст који се прими из `\type` ће у финалном документу да се прикаже фонтом фиксне ширине који је типичан за компјутерске терминале и писаће машине.

Обично бисмо текст који `\type` треба да прикаже окружили витичастим заградама. Међутим, када тај текст и сам садржи отварајуће или затварајуће витичасте заграде, уместо у њих, текст морамо да окружимо са два иста карактера која нису део текста који чини аргумент команде `\type`. На пример: `\type*{*,}` или `\type+{+,}`.

Ако неки од ових карактера грешком употребимо директно, али не за оно за шта је предвиђен, него зато што смо заборавили да резервисани карактер не може да се користи као обичан, могу да се догоде три ствари:

1. Најчешће, грешка приликом компајлирања.
2. Добијемо неочекивани резултат. Ово се дешава посебно са „~” и „%”; у првом случају, уместо „~” који смо очекивали у финалном документу, уметнуће се размак; а у другом случају, све након

карактера „%” у тој линији ће престати да се обрађује. Неправилна употреба „\” такође може да има неочекивани резултат ако он или карактери непосредно иза творе команду коју ConTeXt разуме. Међутим, неправилна употреба „\” много чешће доводи до грешке у компајлирању.

3. Не долази до проблема: ово се дешава са три резервисана карактера који се углавном користе у математици ($_ \wedge \&$): ако се користе ван овог окружења, они се третирају као обични карактери.



Трећа тачка је мој закључак. Истина је да нигде у ConTeXt документацији нисам пронашао где ове резервисане карактере можемо директно да користимо; међутим, у мојим тестовима нисам наишао на грешку када се ово уради; за разлику од, на пример, система L^AT_EX.

3.2 Саме команде

Саме команде увек почињу карактером „\”. У зависности од онога што долази непосредно иза означавајућег низа, прави се разлика између:

- а. **Контролних симбола.** Контролни симбол почиње означавајућим низом („\”) и састоји се само од карактера који нису слова, као на пример, „\”, „\1”, „\’” или „\%”. Контролни симбол може бити било који карактер или симбол који није слово у буквалном смислу, укључујући бројеве, знаке интерпункције, симболе, па чак и размак. Када је потребно да се у овом документу истакне присуство размака (празног простора), користим симбол `_`. Уствари, као што ћемо ускоро да видимо, „_” (обрнута коса црта након које следи размак) је контролни симбол који се често користи, што ћемо ускоро видети.

Празнина или размак је „невидљиви” карактер који у документима као што је овај представља проблем када постоји потреба да се јасно наведе шта треба да се упише у изворни фајл. Кнут је и сам био свестан проблема, па је у својој књизи „The TeXBook” увео обичај представљања значајних размака симболом „_”. Дакле, када је хтео да покаже да су две речи у изворном фајлу раздвојени са два размака, онда је писао „реч1_ реч2”.

- б. **Контролних речи.** Ако је карактер који следи непосредно иза обрнуте косе црте слово у ужем смислу, команда ће представљати *Контролну реч*. Ова група команди је најбројнија од свих и њена особина је да имена команди смеју да се састоје само од слова; бројеви, знаци интерпункције или било који други симболи нису дозвољени. Само мала и велика слова. Имајте на уму да ConTeXt прави разлику између малих и великих слова, што значи да су `\mojakomanda` и `\MojaKomanda` две различите команде. Али би се сматрало да су `\MojaKomanda1` и `\MojaKomanda2` исте команде, јер како ‘1’ и ‘2’ нису слова, они нису део имена команде.



ConTeXt референтно упутство не садржи правила у вези имена команди, као ни остала „упутства” која су део „ConTeXt Standalone”. Оно што сам навео у претходном пасусу је мој закључак заснован на ономе што се дешава у TeX (где се, узгред, акцентовани карактери који нису део енглеског алфабета не сматрају за „слова”). Ово правило омогућава да се на добар начин објасни нестајање размака након имена команде.

Када ConTeXt чита изворни фајл и наиђе на означавајући карактер („\”), он зна да следи команда. Затим чита први карактер након означавајућег низа. Ако он није слово, то значи да је команда контролни симбол и да се састоји само од тог првог симбола. Али с друге стране, ако је први карактер након означавајућег низа слово, ConTeXt ће онда наставити да чита сваки наредни карактер све док не наиђе на први који није слово, па онда зна да је име команде завршено. Ово је разлог што имена команди која су контролне речи не могу да садрже карактере који нису слова.

Када је „не-слово” на крају имена команде размак, претпоставља се да тај размак није део текста који треба да се обради, већ да је уметнут само да се назначи крај имена команде, тако да ConTeXt овај размак. Резултат тога изненађује ConTeXt почетнике, јер када је ефекат команде да се нешто пише у финални документ, писани излаз команде се везује за наредну реч. На пример, следеће две реченице у изворном фајлу

Познавање \TeX помаже да се научи \ConTeXt.

Познавање \TeX, мада није обавезно, помаже да се научи \ConTeXt

редом производе следеће резултате:

Познавање \TeX помаже да се научи ConTeXt.

Познавање \TeX , мада није обавезно, помаже да се научи ConTeXt.¹

Приметите како је у првом случају реч „ \TeX ” спојена са речи која јој следи, али у другом случају није. То је зато што је у првом случају прво „не-слово” након имена команде \TeX био размак, па је потиснут јер ConTeXt претпоставља да се ту налази само да укаже на крај имена команде, док се у другом случају ту налази запета, а пошто то није размак, не потискује се.

С друге стране, овај проблем се не решава једноставним уметањем додатног размака и писањем, на пример,

Познавање $\TeX_{\text{}}_{\text{}}$ помаже да се научи ConTeXt².

неће решити проблем, јер ConTeXt правило (које ћемо видети у одељку 4.2.1) каже да размак апсорбује све празнине и табулаторе који се налазе иза њега. Дакле, када имамо овај проблем (који се на сву срећу не јавља доста често) морамо обезбедити да прво „не-слово” након имена команде није размак. За то постоје два кандидата:

- Резервисани карактери „{” и „}”. Резервисани карактер „{”, као што сам напоменуо, отвара групу, а „}” затвара групу, тако да низ „{” уводи празну групу. Празна група нема ефекта на финални документ, али помаже да ConTeXt зна да име команде испред ње завршило. Или би исто тако могли да креирамо групу око команде у питању, на пример тако што напишемо „{\TeX}”. У сваком случају, резултат ће бити да прво „не-слово” након \TeX није размак.
- Контролни симбол „\” (обрнута коса црта иза које следи размак, погледајте напомену на [страни 35](#)). Овај контролни симбол умеће размак у финални документ. Да би се исправно схватила логика система ConTeXt, требало би да се одвоји мало времена и да се види шта се дешава када ConTeXt наиђе на контролну реч (на пример \TeX) иза које следи контролни симбол (нпр. „\”):
 - ConTeXt наилази на карактер \ иза којег следи „T” и како зна да ово долази испред контролне речи, он наставља са читањем карактера све док не наиђе на „не-слово”, тј. када наиђе на \ карактер који уводи наредни контролни симбол.
 - Једном када сазна да је име команде \TeX , покреће команду и штампа \TeX у финални документ. Затим се враћа на место где је прекинуо читање да провери који карактер следи непосредно након друге обрнуте косе црте.
 - Проверава да је то размак, тј. „не-слово”, што значи да је контролни низ тачно то, тако да може да га изврши. Он то ради и умеће размак.
 - На крају, још једном се враћа на место где је прекинуо читање (размак који је био контролни симбол) и наставља да обрађује остатак изворног фајла.

Овај механизам сам објаснио детаљније, јер елиминација размака често изненађује почетнике. Међутим, треба имати на уму да је проблем релативно мали, јер се контролне речи обично не пишу директно у финални документ, већ утичу на формати и изглед. За разлику од тога, прилично је често да контролни симболи штампају нешто у финални документ.

Постоји и трећа процедура за спречавање проблема размака, која се састоји у дефинисању (у \TeX стилу) сличне команде и укључивањем „не-слова” на крај имена команде. На пример, следећи низ:

```
\def\txt-{\TeX}
```

би креирао команду под именом txt- , која би радила исто као и команда \TeX и функционисала би као треба само ако се позове са цртицом на крају txt- . Технички, ова цртица није део имена команде, али команда неће радити ако се иза имена не стави цртица. Разлог зашто је то тако се тиче механизма за дефинисање \TeX макроа, а то је превише компликовано да се овде објасни. Али функционише: када се ова команда дефинише, сваки пут када употребимо txt- , ConTeXt је замењује са \TeX тако што елиминише цртицу, али користећи је интерно да зна да се име команде завршило, тако да се размак непосредно након команде не брише.

¹ Напомена: у случајевима када у овом уводу треба да се нешто илуструје, писаће се фрагмент кода као и резултат његовог компајлирања користећи једну од две конвенције: понекад су код и резултат његовог компајлирања постављени један уз други у пасусу који се састоји из две колоне; понекад се код испишује тамно магента нијансом којом се у овом документу углавном представљају ConTeXt команде, а резултат његовог компајлирања у црвеној боји.

² У вези симбола „\”, присетите се напомене на [страни 35](#).

Овај ‘трик’ неће радити како треба са `\define` командом, ConTeXt команда за дефинисање макроа.

3.3 Опсег важења команди

3.3.1 Комадне којима треба или не треба назначити опсег важења

Многе ConTeXt команде, посебно оне које утичу на форматирање особина фонтова (црни слог, курзив, капитал, итд.), укључују одређену особину која остаје активна све док се не наиђе на наредну команду која је искључује, или која укључује неку другу особину која није компатибилна са њом. На пример, команда `\bf` укључује црни слог, и он остаје активан све док се не наиђе на *некомпатибилну* команду, као што је `\tf`, или `\it`.

Пошто нису дизајниране да се примене само на одређени текст, није потребно да ове врсте команди узимају било какав аргумент. Исто је као да су ограничене само на *укључивање* некакве функције (црни слог, курзив, безсерифна слова, одређена величина фонта, итд.)

Када се ове команде изврше унутар *групе* (погледајте [одељак 3.8.1](#)), оне губе свој ефекат када се група у којој се изврше затвори. Стога, да би ове команде утицале само на одређени део текста, често се генерише група која садржи команду и текст на који желимо да она утиче. Група се креира тако што се њен садржај окружи витичастим заградама. Дакле, следећи текст

```
У {\it The \TeX Book}, {\sc Кнут}
је објаснио све што треба да знате о
\TeX.
```

```
У The TeXBook, Кнут је објаснио све што треба да знате о TeX.
```

креира две групе, једну која одређује опсег важења `\it` (курзив) команде и друге која одређује опсег `\sc` (капитал) команде.

За разлику од ове врсте команде, постоје и остале које услед ефекта који имају или неких других разлога, захтевају директну навођење текста на који треба да се примене. У тим случајевима се текст на који команда треба да се примени поставља унутар витичастих заграда *непосредно након команде*. Као пример овога би могли да поменемо команду `\framed`: ова команда исцртава оквир око текста који јој се прослеђује као аргумент, тако да

```
\framed{Татамата и Дедабрада}
```

ће произвести

```
Татамата и Дедабрада
```

Приметите да иако се у првој групи команди (онима које захтевају аргумент) витичасте заграде понекад користе и за одређивање опсега дејства, то није неопходно да би команда функционисала. Команда је дизајнирана да се примени од места на којем се појави. Зато, када се заградама одређује поље њене примене, команда се поставља *унутар ових заграда*, за разлику од друге групе команди, код којих заграде које окружују текст над којем треба да се примени команда, долазе *након* команде.

У случају `\framed` команде, очигледно је да је за њен ефекат неопходан аргумент – текст на који треба да се примени. У осталим случајевима, од програмера зависи да ли је команда једног или другог типа. Тако, на пример, оно што раде команде `\it` и `\color` је прилично слично: оне на текст примењују особину (формат или боју). Али донета је одлука да се прва програмира без аргумената, а друга као команда са аргументом.

3.3.2 Команде којима је потребно изричито навођење где почињу и где се завршавају (окружења)

Постоје одређене команде које свој опсег важења одређују прецизним назначивањем места на којем треба да почну и места на којем њихово дејство престаје. Тако да ове команде долазе у паровима:

једна означава када се команда укључује, а друга када њено дејство треба да престане. „start”, након којег долази име команде се користи да означи почетак акције, а „stop”, након којег такође долази име команде, назначава крај. Тако на пример команда „itemize” постаје `\startitemize` и означава почетак *набрајања*, а `\stopitemize` означава место где престаје набрајање.

За ове парове команди не постоји посебно име у званичној ConTeXt документацији. Референтно упутство и увод их просто зову „start ... stop”. Понекад се називају *окружења*, што је име које L^AT_EX даје сличној врсти конструкција, мада ово има недостатак јер се у систему ConTeXt израз „окружење” користи за нешто сасвим друго (посебну врсту фајла о којем ћемо говорити када у одељку 4.6 будемо причали о пројектима у више фајлова). Чак штавише, пошто је израз окружење јасан, а из контекста ће бити лако да се направи разлика да ли говоримо о *командама окружења* или *фајловима окружења*, ја ћу употребљавати овај израз.

Окружења се, дакле, састоје из команде која их отвара или започиње, и друге које их затвара или завршава. Ако изворни фајл садржи команду за отварање окружења које се касније не затвара, обично ће се генерисати грешка.¹ С друге стране, ове врсте грешака се теже проналазе, јер грешка може да се појави много даље иза места на којем се налази команда отварања. Понекад ће нам „.log” фајл приказати линију у којој почиње окружење које није правилно затворено; али у другим приликама ће недостатак затварања окружења значити да ConTeXt погрешно интерпретира одређени пасаж и не у том неисправном окружењу, што значи да „.log” фајл и није од неке помоћи у откривању места проблема.

Окружења могу да се угнезде, што значи да друго окружење може да се отвори унутар постојећег окружења, мада у случајевима када постоје угњеждена окружења, окружење мора да се затвори унутар окружења у којем је било отворено. Другим речима, редослед у којем се окружења затварају мора бити конзистентан са редоследом у којем су била отворена. Верујем да би ово требало да буде јасно из следећег примера:

```
\startNesto
...
\startNestoDrugo
...
\startJosNestoDrugo
...
\stopJosNestoDrugo
\stopNestoDrugo
\stopNesto
```

У примеру можете видети како је окружење „JosNestoDrugo” отворено унутар окружења „NestoDrugo”, па мора унутар њега и да се затвори. Ако се то уради на неки други начин, генерисаће се грешка приликом компајлирања.

У општем случају, команде дизајниране као *окружења* су оне које имплементирају неку измену која треба да се примени на јединице текста не мање од пасуса. На пример, окружење „narrower” које мења маргине има смисла само када се примени на нивоу пасуса; или окружење „framedtext” које исцртава оквир око једног или више пасуса. Ово последње окружење нам може помоћи да разумемо зашто су неке команде дизајниране као окружења, а неке као индивидуалне команде: ако желимо да уоквиримо једну или више речи у једној линији, употребимо команду `\framed`, али ако желимо да уоквиримо цео пасус (или неколико пасуса) онда ћемо употребити „framedtext” окружење.

С друге стране, текст који се налази унутар одређеног окружења обично чини *групу* (погледајте одељак 3.8.1), што значи да ако се унутар окружења пронађе активациона команда, за разлику од свих команди које се примењују на сав текст који следи, ова команда ће се примењивати само до краја окружења у којем се пронађе; а ConTeXt уствари има неименовано *окружење* које почиње командом `\start` (без икаквог текста иза; просто *start*. Зато га и зовем *неименовано окружење*) и завршава се командом `\stop`. Чини ми се да је једина његова функција да креира групу.

¹ Мада не и увек; то зависи од врсте окружења и од ситуације у остатку документа. ConTeXt се по овом питању разликује од система L^AT_EX, који је много стриктнији.



Нигде у ConTeXt документацији нисам прочитао да је један од ефеката окружења да групишу свој садржај, али то је резултат мојих текстова са више предефинисаних окружења, мада морам признати да ти тестови нису били превише исцрпни. Просто сам проверио нека насумично изабрана окружења. Међутим, моји тестови показују да би таква тврдња, у случају да је истинита, важила само за нека предефинисана окружења: она креирана командом `\definestartstop` (која се објашњава у одељку 3.7.2) не креирају било какву групу, осим када током дефинисања новог окружења наведемо и команде које су потребне да се креира група (погледајте одељак 3.8.1).

Такође је само моја претпоставка да окружења која сам назвао *неименована* (`\start`) окружења постоје само да би се креирала група: она креирају групу, али не знам да ли имају још неку другу примену. Ово је једна од недокументованих команди у референтном упутству.

3.4 Опције рада команде

3.4.1 Команде које могу да раде на неколико различитих начина

Многе команде могу да раде на више од једног начина. У тим случајевима увек постоји предодређени начин функционисања који се може променити навођењем параметара који одговарају жељеном начину рада у заградама након имена команде.

Добар пример овога што сам рекао налазимо у команди `\framed` поменутој у претходном одељку. Ова команда црта оквир око текста који јој се прослеђује као аргумент. Подразумевано је да оквир има висину и ширину текста на који се примењује; али можемо да наведемо и неку другу висину о ширину. Тако да можемо видети разлику између функционисања подразумеване `\framed` команде:

```
\framed{Татамата}
```

и прилагођене верзије функције:

```
\framed
[width=3cm, height=1cm]
{Татамата}
```

У другом примеру смо унутар великих заграда навели одређену ширину и висину оквира који окружује текст који команда узима као аргумент. Унутар заграда се различите конфигурационе опције раздвајају запетама; размаци па чак и преломи линије (све док нису двоструки прелом линије) између две или више опција се не узимају у обзир, тако да на пример, наредне четири верзије исте команде дају потпуно исти резултат:

```
\framed[width=3cm,height=1cm]{Татамата}

\framed[width=3cm, height=1cm]{Татамата}

\framed
[width=3cm, height=1cm]
{Татамата}

\framed
[width=3cm,
 height=1cm]
{Татамата}
```

Очигледно је да се последња верзија најлакше чита: на први поглед можемо видети колико има опција и како се користе. У примеру као што је овај, са само неколико опција, то можда и не изгледа тако важно; али у случајевима када постоји дугачка листа опција, ако свака од њих заузима посебну линију у изворном фајлу, лакше ће се разумети шта изворни фајл тражи да ConTeXt уради, а такође, ако је то неопходно, лакше ће се пронаћи и потенцијална грешка. Зато је 'пожељно' да корисници употребљавају овај последњи формат (или неки сличан) писања команди.

Што се тиче синтаксе конфигурационих опција, погледајте даље у тексту (одељку 3.5).

3.4.2 Команде које конфигуришу начин на који раде друге команде (`\setupNesto`)

Већ смо видели да команде које имају различите начине функционисања увек поседују подразумевани начин на који раде. Ако се нека од ових команди позива више пута у изворном фајлу, а желимо да изменимо подразумевано понашање сваке од њих, уместо да приликом сваког позива мењамо те опције, много је zgodније и ефикасније да се промени подразумевано подешавање. За то је скоро увек доступна команда чије име почиње са `\setup`, након чега следи име команде чија подразумевана подешавања желимо да променимо.

Команда `\framed` коју смо као пример користили у овом одељку и даље служи као добар пример. Дакле, ако у свом документу имамо доста оквира, али сваки од њих захтева прецизне димензије, онда је најбоље да помоћу `\setupframed` реконфигуришемо начин на који функционише команда `\framed`. Тако ће

```
\setupframed
[
  width=3cm,
  height=1cm
]
```

обезбедити да од сада сваки позив команде `\framed` подразумевано прави оквир ширине 3 и висине 1 центиметар, без потребе да се то наводи сваки пут.

Постоји неких 300 ConTeXt команди које нам омогућавају да конфигуришемо начин функционисања осталих команди. Тако да можемо конфигурисати подразумевано понашање оквира (`\framed`), листи („`itemize`”), наслова поглавља (`\chapter`), или наслова одељака (`\section`), итд.

3.4.3 Подешавање прилагођених верзија подесивих команди (`\defineNesto`)

Ако наставимо са `\framed` примером, постаје очигледно да ако наш документ користи неколико врста оквира, сваки са другачијим димензијама, онда би било идеално када би могли да *предефинишемо* различите конфигурације команде `\framed`, па да им придружимо одређено име којим сваку од њих можемо по потреби да користимо. У систему ConTeXt то можемо да урадимо `\defineframed` командом, која има следећу синтаксу:

```
\defineframed[Име][Конфигурација]
```

где *Име* представља име додељено одређеној врсти оквира који се конфигурише; а *Конфигурација* је одређена конфигурација придружена том имену.

Ефекат свега овога је да ће се наведена конфигурација придружити имену које смо јој доделили, и које ће се за све намене и сврхе понашати као да је нека нова команда. Можемо да га користимо у било ком контексту у којем би могли да користимо и оригиналну команду (`\framed`).

Ова могућност не постоји само за овај конкретни случај команде `\framed`, већ и за многе команде које поседују `\setup` могућност. Комбинација `\defineNesto` + `\setupNesto` је механизам који систему ConTeXt даје његову екстремну снагу и флексибилност. Ако детаљно истражимо шта ради команда `\defineNesto`, видећемо да:

- Она најпре клонира одређену команду која подржава мноштво конфигурација.
- Тај клон придружује имену нове команде.
- Коначно, поставља предодређену конфигурацију клона која се разликује од начина на који је била конфигурисана оригинална команда.

У примеру који смо навели, конфигурисали смо наш специјални оквир у време када смо га креирали. Али исто тако прво можемо да га креирамо, па да га касније конфигуришемо, јер као што сам напоменуо, једном када се креира клон, он може да се користи свуда где може да се користи и оригинал. Тако на пример, ако смо креирали оквир под именом „MojSpecijalniOkvir“, можемо да га конфигуришемо командом `\setupframed` уз навођење одређеног оквира који желимо да конфигуришемо. У овом случају ће `\setup` команда да узме нови аргумент који представља име оквира који се конфигурише:

```
\defineframed[MojSpecijalniOkvir]
\setupframed
[MojSpecijalniOkvir]
[ ... ]
```

3.5 Резиме синтаксе команде и опција, употреба великих и витичастих заграда приликом позивања

Ево резимеа онога што смо видели до сада. У систему ConTeXt

- Команде у ужем смислу увек почињу карактером „\“.
- Неке команде могу да узму један или неколико аргумената.
- Аргументи који команди говоре *како* да функционише или који на неки начин утичу на функционисање команде се наводе унутар великих заграда.
- Аргументи који команди говоре над којим делом текста треба да делује се наводе унутар витичастих заграда.

Када команда да делује само над једним словом, као што је на пример то случај са командом `\buildtextcedilla` (само као пример – ‘ç’ се често користи у каталонском и помало у француском језику), витичасте заграде око аргумента могу да се изоставе: команда ће се применити на први карактер који није размак.

- Неки аргументи нису обавезни, па у том случају можемо да их изоставимо. Али оно што не можемо никада да променимо јесте редослед аргумената који команда очекује.

Аргументи који се наводе унутар великих заграда могу бити разних врста. Углавном:

- Могу имати само једну вредност која је скоро увек нека реч или фраза.
- Могу имати разне опције, па у том случају могу:
 - Да се представе само једном речи која би могла да буде симболичко име (оно чије значење ConTeXt познаје), мера или димензија, број, име неке друге команде, итд.
 - Да се састоје од имена променљивих којима мора бити задата вредност. У овом случају нам званична дефиниција команде (погледајте [одељак 3.6](#)) увек говори коју врсту вредност очекује свака од опција.
- * Када је очекивана вредност опције текст, он може да садржи размаке, а такође и команде. У тим случајевима је понекад zgodno да се вредност постави унутар витичастих заграда.
- * Када је вредност коју очекује опција команда, обично можемо да наведемо више команди као ту вредност, мада је понекад потребно да све команде које представљају вредност опције поставимо у витичасте заграде. Вредност опције такође морамо да поставимо у витичасте заграде када било која од команди које су део вредности узимају опцију унутар великих заграда.

У оба случаја ће различите опције које узима исти аргумент бити раздвојене запетама. Размаци и преломи линија (сви осим двоструких прелома) између различитих опција се игноришу. Такође се игноришу и размаци и преломи линија између различитих аргумената.

- Коначно, у систему ConTeXt се никада не догађа да исти аргумент истовремено добија опције које са састоје од речи и опције које се састоје од променљиве којој се вредност експлицитно додељује. Другим речима, можемо имати овакву опцију

```
\komanda[Opcija, Opcija2, ...]
```

а неку другу као што је

```
\komanda[Promenljiva1=vrednost, Promenljiva2=vrednost, ...]
```

Али никада нећемо наићи на мешавину обе:

```
\komanda[Opcija1, Promenljiva1=vrednost, ...]
```

3.6 Званична листа ConTeXt команди

У ConTeXt документацији постоји један посебно важан документ који приказује листу свих команди и за сваку од њих наводи колико аргумената које врсте очекује, као и различите осмишљене опције и њихове дозвољене вредности. Овај документ се назива „setup-en.pdf” и аутоматски се генерише за сваку нову верзију система ConTeXt. Можете га пронаћи у директоријуму „tex/texmf-context/doc/context/documents/general/qrcs”.

Уствари постоји седам верзија „qrcs” документа, по једна за сваки од језика који имају ConTeXt интерфејс: немачки, чешки, француски, холандски, енглески, италијански и румунски. За сваки од ових језика у директоријуму постоје два документа: један се назива „setup-КодЈезика” (где је КодЈезика двословни међународни идентификатор језика), а други се зове „setup-mapping-КодЈезика”. Овај други документ садржи листу команди у абecedном редоследу која наводи *прототип* команде, али без даљих информација омогућим вредностима за сваки аргумент.

Овај документ је од изузетног значаја за учење употребе система ConTeXt, јер у њему можемо да пронађемо да ли одређена команда постоји или не; ово је заиста корисно, имајући у виду комбинацију команда (или окружење) + setupкоманда + defineкоманда. На пример, ако знам да се празна линија уноси командом `\blank`, Могу да пронађем да ли постоји команда под називом `\setupblank` која ми омогућава да је конфигуришем, и друга која ми дозвољава да подезим прилагођену конфигурацију за празне линије, (`\defineblank`).

„setup-en.pdf” је дакле, незаменљив код учења система ConTeXt. Али ја бих заиста више волео, да нам он првенствено каже да ли команда ради само у верзији Mark II или Mark IV, а посебно да нам уместо набрајања типова аргумената које узима свака команда каже чему служе ти аргументи. То би у знатној мери умањило недостатке ConTeXt документације. Неке команде дозвољавају и необавезне аргументе које у овом уводу чак ни не помињем јер не знам чему служе, а пошто нису обавезни нема било какве потребе да их помињем. Ово заиста изазива огромну фрустрацију.

3.7 Дефинисање нових команди

3.7.1 Општи механизам за дефинисање нових команди

Управо смо видели како помоћу `\defineNesto` можемо да клонирамо постојећу команду и развијемо њену нову верзију, која ће за све намене и сврхе функционисати као нова команда.

Уз ову могућност, која је доступна само за неке одређене команде (сигурно поприличан број њих, али не све), ConTeXt поседује општи механизам за дефинисање нових команди који је изузетно моћан,

мада је у одређеним применама и прилично сложен. У тексту као што је овај, намењеном почетницима, мислим да је најбоље да се уведе тако што се почне од неких његових најједноставнијих употреба. Најједноставнија од свих је да се фрагменти текста придруже некој речи, па сваки пут када се та реч појави у изворном фајлу, она се замени са текстом повезаним са њом. С једне стране, ово ће нам омогућити да уштедимо доста времена за куцање, а с друге стране, додатна предност је што се умањује могућност грешака у куцању, а у исто време се обезбеђује да је тај текст увек исписан на исти начин.

Хајде да замислимо, на пример, да пишемо трактат о алитерацији у латинским текстовима, у којем често цитирамо латинску реченицу „*O Tite tute Tati tibi tanta tyranne tulisti*” (О Тите Татијусе, ти тиранине, толико тога си наукао на себе!). То је прилично дугачка реченица у којој су две речи личне именице које почињу великим словом и где је, морамо признати, колико год да волимо латинску поезију, лако да се „саплетемо” када је пишемо. У овом случају, могли бисмо да у преамбулу изворног фајла једноставно ставимо следеће:

```
\define\Tite{\quotation{O Tite tute Tati tibi tanta tyranne tulisti}}
```

Према оваквој дефиницији, сваки пут када се команда `\Tite` појави у нашем изворном фајлу, она ће се заменити за наведеном реченицом, а налазиће се и унутар знака навода као што их је имала и оригинална дефиниција, чиме смо обезбедили да ће реченица сваки пут имати исти изглед. Могли смо и да је испишемо курзивом, већом величином фонта... како год желимо. Битна ствар је што морамо да је испишемо само једном, а кроз цео текст ће се исписивати на потпуно исти начин на који смо је написали, небитно колико често се понављала. Такође бисмо могли да креирамо две верзије команде под називом `\Tite` и `\tite`, у зависности од тога да ли је потребно да се реченица испише великим словима или не. Текст замене може бити чист текст, а може да садржи и команде, или да формира математичке изразе у којима је (барем у мом случају) већа вероватноћа да дође до грешке у куцању. На пример, ако је потребно да се израз (x_1, \dots, x_n) често појављује у нашем тексту, могли би да креирамо команду која га представља. На пример

```
\define\xvec{$(x_1,\ldots\thinspace,x_n)$}
```

тако да кад год се у тексту појави `\xvec`, замениће се са изразом који је придружен тој команди.

У општем случају, синтакса `\define` команде је:

```
\define[БрАргумента]\ИмеКоманде{ТекстЗаЗамену}
```

где се

- **БрАргумента** односи на број аргумената које узима нова команда. Ако не узима ниједан, као у претходним примерима, ово може и да се изостави.
- **ИмеКоманде** односи на име које се даје новој команди. Овде важи опште правило које се тиче имена команди. Име би могло да буде и један карактер који није слово, или једно или више слова без употребе било каквог „не-слово” карактера.
- **ТекстЗаЗамену** састоји из текста који ће да замени име нове команде, сваки пут када се она појави у изворном фајлу.

Могућност дефинисања нових команди са аргументима у дефиницији даје механизму велику флексибилност, јер омогућава да се дефинише променљиви текст замене који ће одредити прослеђени аргументи.

На пример: замислимо да желимо написати команду која креира почетак пословног писма. Врло проста верзија тога би била:

```
\define\ZaglavljjePisma{
  \rightaligned{Петар Петровић}\par
  \rightaligned{Консултант}\par
  Зајечар, \date\par
  Драги господине,\par
}
```

али би било добро да имамо верзију команде која би у заглављу исписала име примаоца. То би захтевало да употребимо параметар који новој команди преноси име примаоца писма. Команда би требало да се редефинише на следећи начин:

```
\define[1]\ZaglavljjePisma{
  \rightaligned{Петар Петровић}\par
  \rightaligned{Консултант}\par
  Зајечар, \date\par
  Драги г. #1,\par
}
```

Приметите да смо у дефиницију унели две измене. Прво смо између кључне речи `\define` и новог имена команде, уметнули 1 у велике заграде ([1]). То систему ConTeXt говори да команда коју дефинишемо узима један аргумент. Даље, у последњој линији дефиниције команде, написали смо „Драги г. #1,“, користећи резервисани карактер „#“. То наводи да се у тексту замене на месту појављивања „#1“ умеће садржај првог аргумента. Да је било два параметра, „#1“ би се односило на први параметар, а „#2“ на други. Да би се (у изворном фајлу) позвала команда, након имена команде морају да се у витичастим заградама наведу аргументи, сваки у свом пару заграда. Тако да би команда коју смо управо дефинисали, требало да се у тексту изворног фајла позива на следећи начин:

```
\ZaglavljjePisma{Име примаоца}
```

На пример: `\ZaglavljjePisma{Марко Марковић}`.

Претходну функцију би још могли да унапредимо, јер претпоставља да ће се писмо слати мушкарцу (поставља „Драги господине“), тако да би требало да се наведе још један параметар којим се прави разлика између мушког и женског примаоца. На пример:

```
\define[2]\ZaglavljjePisma{
  \rightaligned{Петар Петровић}\par
  \rightaligned{Консултант}\par
  Зајечар, \date\par
  #1\ #2,\par
}
```

тако да би функција могла да се позове, на пример, са

```
\ZaglavljjePisma{Драга г.}{Марија Марјановић}
```

мада ово и није баш елегантно (са програмерске тачке гледишта). Боље би било да се симболичке вредности дефинишу као први аргумент (мушкарац/жена; 0/1; м/ж) тако да сам макро према тој вредности изабере одговарајући текст. Али да бисмо објаснили како се то постиже, потребно је да уђемо дубље у материју коју сматрам да читалац-почетник у овом тренутку не може да разуме.

3.7.2 Креирање нових окружења

За креирање новог окружења, ConTeXt обезбеђује `\definestartstop` команду која има следећу синтаксу:

```
\definestartstop[Име][Опције]
```



У званичној `\definestartstop` дефиницији (погледајте [одељак 3.6](#)) постоји додатни аргумент који овде нисам навео, и за који нисам успео да пронађем чему служи. Не објашњавају га ни уводни ConTeXt „Excursion“, ни недовршено референтно упутство. Претпоставио сам да би овај аргумент (који мора да се унесе између имена и конфигурације) могао да буде име неког постојећег окружења које би послужило као почетни модел за ново окружење, али моји тестови показују да је та претпоставка погрешна. Претражио сам ConTeXt мејлинг листу и нисам наишао на било какву употребу овог могућег аргумента.

где

- **Име** представља име које ће добити ново окружење.
- **Конфигурација** омогућава да конфигуришемо понашање новог окружења. Имамо на располагању следеће вредности којима можемо да га конфигуришемо:

- `before` – Команде које треба да се изврше пре уласка у окружење.
- `after` – Команде које треба да се изврше након напуштања окружења.
- `style` – Стил који мора да има текст новог окружења.
- `setups` – Скуп команди креиран са `\startsetups ... \stopsetups`. Овај увод не објашњава ову команду и њену употребу.
- `color`, `inbetween`, `left`, `right` – Недокументоване опције које нисам успео да употребим. Из њиховог имена можемо претпоставити шта неке од њих раде, на пример `color`, али из више тестова које сам извршио, дајући неку вредност тој опцији, нисам видео било какву промену унутар окружења.



Ево примера могуће дефиниције окружења:

```
\definestartstop
[TextWithBar]
[before=\startmarginrule\noindeatation,
 after=\stopmarginrule,
 style=\ss\sl
]

\starttext

Прва два основна закона људске глупости недвосмислено тврде да:

\startTextWithBar

\starttitemize[n,broad]

\item Увек и неизбежно потцењујемо број глупих индивидуа на свету.

\item Вероватноћа да је дата особа глупа не зависи од било које
друге карактеристике те исте особе.

\stoptitemize

\stopTextWithBar

\stoptext
```

Резултат би био:

Прва два основна закона људске глупости недвосмислено тврде да:

1. *Увек и неизбежно потцењујемо број глупих индивидуа на свету.*
2. *Вероватноћа да је дата особа глупа не зависи од било које друге карактеристике те исте особе.*

Ако желимо да наше ново окружење буде група (одељак 3.8.1), тако да било каква измена уобичајеног начина функционисања система ConTeXt која се деси унутар окружења престане чим се напусти окружење, морамо да укључимо команду `\bgroup` у опцију „before” и команду `\egroup` у опцију „after”.

3.8 Остали основни концепти

Осим команди, постоје и други појмови који су фундаментални за разумевање логике иза начина на који функционише систем ConTeXt. Због своје сложености, неки од њих нису погодни за увод, па у

овом документу о њима нећемо говорити; али постоје два појма која би сада требало да испитамо: групе и димензије.

3.8.1 Групе

Група је добро дефинисан фрагмент изворног фајла који ConTeXt користи као *радну јединицу* (ускоро ћу објаснити шта то значи). Свака група има почетак и крај који експлицитно морају да се назначе. Група почиње:

- Резервисаним карактером „{” или командом `\bgroup`.
- Командом `\begingroup`
- Командом `\start`
- Отварањем одређених окружења (командом `\startNesto`).
- Отварањем математичког окружења (резервисаним карактером „\$”).

а затвара се

- Резервисаним карактером „}” или командом `\egroup`.
- Командом `\endgroup`
- Командом `\stop`
- Затварањем окружења (командом `\stopNesto`).
- Напуштањем математичког окружења (резервисаним карактером „\$”).

Одређене команде такође аутоматски генеришу групу, на пример, `\hbox`, `\vbox` и, у општем случају, команде повезане са креирањем *кутија*¹. Осим ових последњих случајева (групе које одређене команде аутоматски генеришу), начин затварања групе мора бити конзистентан са начином на који је отворена. Ово значи да група која је започета са „{” мора да се затвори са „}”, а група започета са `\begingroup` мора да се затвори са `\endgroup`. Ово правило има само један изузетак, да група започета са „{” може да се затвори са `\egroup`, а да група започета са `\bgroup` може да се затвори са „}”; то у суштини значи да су „{” и `\bgroup` потпуно идентични и могу да се користе једно уместо другог, а слично је и са „}” и `\egroup`.

Команде `\bgroup` и `\egroup` су дизајниране да би били у могућности да отворимо и затворимо групу. Стога, из разлога који се тичу TeX синтаксе, те групе нису могле да отворе и затворе витичастим заградама, јер би се тако у изворном фајлу генерисале неупарене витичасте заграде, а то би увек изазивало грешку током компајлирања.

За разлику од њих, команде `\begingroup` и `\endgroup` не могу да се користе уместо витичастих заграда или `\bgroup ... \egroup` команди јер група која је започета са `\begingroup` мора да се затвори са `\endgroup`. Ове друге команде су дизајниране тако да се омогући детаљнија провера грешака. У општем случају, обични корисници немају потребу да их користе.

Можемо имати угњеждене групе (групу унутар друге групе), па у том случају редослед у којем се групе затварају мора бити конзистентан са редоследом у којем су биле отворене: било која подгрупа мора да се затвори унутар групе у којој је била отворена. Такође могу да постоје и празне групе, генерисане са „{ }”. У принципу, празна група нема утицаја на финални документ, али може бити корисна, на пример, да се назначи крај имена команде.

Главна намена група је да обухвате свој садржај у једну целину: дефиниције, формати и доделе вредности које се ураде унутар групе се по правилу „заборављају” чим напустимо групу. Дакле, ако желимо да ConTeXt привремено измени свој уобичајени начин функционисања, најбољи начин да то постигнемо је да креирамо групу, па да унутар ње изменимо то функционисање. Затим, када напустимо групу, све вредности и формати који су важали пре ње ће се вратити. Већ смо видели неке примере овога када смо поменули команде као што су `\it`, `\bf`, `\sc`, итд. Али ово се не дешава само са командама форматирања: група на неки начин *изолује* свој садржај, тако да било каква измена било које од многих интерних променљивих којима ConTeXt константно управља важи само док се налазимо унутар групе у којој је промена направљена. Слично, команда дефинисана унутар групе не постоји ван ње.

¹ Појам *кутија* је такође основни ConTeXt појам, али се његово објашњење не даје у овом уводу.

Тако да ако обрадимо следећи пример

```
\define\A{B}
\A
{
  \define\A{C}
  \A
}
```

видећемо да када се први пут изврши команда `\A`, резултат одговара њеној почетној дефиницији ('B'). Затим смо креирали групу и у њој редефинисали команду `\A`. Ако је сада извршимо унутар групе, команда ће нам вратити нову дефиницију ('C' у нашем примеру), али када напустимо групу у којој је команда `\A` била редефинисана и извршимо је још једном, она ће поново да испише 'B'. Дефиниција направљена унутар групе се „заборавља” чим напустимо ту групу.

Још једна могућа употреба група се тиче команди или инструкција дефинисаних да се примене само на карактер написан иза њих. У овом случају, ако желимо да се команда примени на више карактера, морамо да их поставимо унутар групе. Тако на пример, резервисани карактер „^” који, као што већ знамо, конвертује наредни карактер у експонент када се користи унутар математичког окружења; па ако на пример напишемо „\$4^2x\$” добићемо „4²x”. Али ако напишемо „\$4^{2x}\$” добићемо „4^{2x}”.

Конечно, трећа употреба груписања је да се систему ConTeXt каже да оно што се налази унутар групе третира као целину. То је разлог што је раније у (одељку 3.5) речено да је у неким ситуацијама боље да се садржај неке опције команде стави унутар витичастих заграда.

3.8.2 Димензије

Мада би систем ConTeXt могли савршено добро да користимо без бриге о димензијама, не бисмо могли да искористимо све могућности подешавања ако се не упознамо са њима. Јер типографска перфекција система TeX и оних изведених из њега у великој мери зависи од тога што систем интерно води рачуна о димензијама. Карактери имају димензије; размак између речи, или између линија, или између пасуса има димензије; линије имају димензије; маргине, заглавља и подножја. Постојаће димензија за скоро сваки елемент стране који може да нам падне на памет.

Димензије се у систему ConTeXt наводе децималним бројем иза кога следи јединица мере. У [табели 3.2](#) су наведене јединице које могу да се употребе.

| Име | Име у ConTeXt | Еквивалент |
|-----------------|---------------|-----------------------|
| Инч | in | 1 in = 2.54 cm |
| Центиметар | cm | 2.54 cm = 1 инч |
| Милиметар | mm | 100 mm = 1 cm |
| Тачка | pt | 72.27 pt = 1 инч |
| Велика тачка | bp | 72 bp = 1 инч |
| Скалирана тачка | sp | 65536 sp = тачка |
| Пика | pc | 1 pc = 12 тачака |
| Дидо | dd | 1157 dd = 1238 тачака |
| Цицero | cc | 1 cc = 12 дидоа |
| | em | |
| | ex | |

Табела 3.2 Јединице мере у систему ConTeXt

Прве три јединице у [табели 3.2](#) су стандардне јединице за дужину; прва се користи у неким деловима света енглеског говорног подручја, а остале ван њега или у неким његовим деловима. Остале јединице долазе из света типографије. Последње две, за које нисам навео еквивалент, су релативне јединице мере чија је основа текући фонт. 1 „em” представља ширину карактера „М”, а „ex” представља ширину карактера „х”. Употреба мера везаних за величину фонта омогућава креирање макроса

који изгледају једнако добро без обзира на то који извор се користи у датом тренутку. Зато се у општем случају и препоручује њихова употреба.

Уз неколико изузетака, можемо да користимо било коју јединицу мере која нам одговара, јер ће ConTeXt интерно да их конвертује. Али увек када се наведе димензија, обавезно је да се наведе и јединица мере, па чак и ако желимо да задамо меру „0“, морамо да напишемо '0pt' или '0cm'. Између броја и имена јединице можемо, али не морамо да уметнемо размак. Ако јединица има децимални део, можемо да употребимо децимални граничник, или (.) или запету (,).

Мере се обично користе као опција неке команде. Али можемо и директно да доделимо вредност некој интерној мери система ConTeXt само је потребно да знамо њено име. На пример, увлачење прве линије пасуса се у систему ConTeXt интерно контролише променљивом која се назива `\parindent`. Ако јој експлицитно наведемо вредност, променићемо од тада па надаље меру коју користи ConTeXt. Па тако, ако желимо да елиминишемо увлачење прве линије, потребно је само да у изворном фајлу напишемо:

```
\parindent=0pt
```

Такође смо могли да напишемо и `\parindent 0pt` (без знака једнакости) или `\parindent0pt` без размака између имена јединице и вредности.

Међутим, сматра се да директна додела интерној мери „није елегантна“. У општем случају, препоручује се да се употребе команде које контролишу ту променљиву, и да се то уради у преамбули изворног фајла. Ако се тако не уради, добија се изворни фајл у којем се грешке врло тешко отклањају јер се све конфигурационе команде не налазе на истом месту, па је заиста тешко да се достигне одређена конзистентност типографских карактеристика.

Неке од димензија које користи ConTeXt су „еластичне“, то јест, у зависности од контекста, оне имају једну или другу вредност. Ове мере се наводе следећом синтаксом:

```
\ИмеМере plus МаксУвећање minus МаксУмањење
```

На пример

```
\parskip 3pt plus 2pt minus 1pt
```

Ова инструкција систему ConTeXt говори да димензији `\parskip` (која задаје вертикални размак између пасуса) *нормалну* меру од 3 тачке, али тако да ако композиција странице то захтева, мера може да буде до 5 тачака (3 плус 2) или само 2 тачке (3 минус 1). У овим случајевима ће ConTeXt изабрати размак за сваку страницу између минимално 2 тачке и максимално 5 тачака.

3.9 Метода за самостално учење система ConTeXt

Испоставља се да је огромна количина ConTeXt команди и опција заиста поражавајућа и може да нам остави утисак да никада нећемо успети да радимо у њему на одговарајући начин. Овај утисак може да заваља, јер је једна од предности систем ConTeXt уједначен начин на који ради са свим својим структурама: Ако добро научимо неколико структура, и ако, мање више знамо чему служе остале, биће нам релативно лако да научимо како се користе када нам буде била потребна нека додатна могућност. Стога овај увод посматрам као на врсту *обуке* која ће да нас приреди за своја сопствена истраживања.

Да бисте креирали документ системом ConTeXt, вероватно је неопходно да познајете само следећих пет основних ствари (могли бисмо их назвати ConTeXt *Ton 5*):

1. Како да креирате изворни фајл или пројекат; ово се објашњава у [поглављу 4](#) овог увода.
2. Поставите главни фонт документа и знате основне команде за промену фонта и боја ([поглавље 6](#)).

3. Основне команде за креирање структуре садржаја документа, као што су поглавља, одељци, по-
додељци, итд. Све ово је објашњено у [поглављу 7](#).
4. Можда и како да управљате окружењем *набрајања*, што је детаљно описано у [одељку 12.3](#).
5. ... још понешто.

Што се тиче осталог, све што је потребно је знати да је могуће. Сигурно је да нико неће користити могућност ако не зна да уопште и постоји. Многе од њих су описане у овом уводу; али првенствено, увек можемо да посматрамо како се систем ConTeXt увек понаша када наиђе на одређену врсту конструкције:

- Прво ће постојати команда која имплементира могућност.
- Друго, скоро увек и команда која нам омогућава да конфигуришемо и унапред одредимо како ће се задатак извршити; команда чије име почиње са `\setup` и која се обично подудара са основном командом.
- Коначно, обично је могуће да се креира нова команда која обавља сличне задатке, али са другом чијом конфигурацијом.

Ако желите сазнати да ли ове команде постоје или не, претражите званичну листу команди (погледајте [одељак 3.6](#)), која ће вас такође упознати са конфигурационим опцијама које могу да се користе са командама. И мада на први поглед имена ових команди могу изгледати *неразумљиво*, убрзо ћемо видети да постоје опције које се понављају у многим командама и које у свим тим командама раде на исти или врло сличан начин. Ако нисмо сигурни у то шта нека опција ради, или како ради, биће довољно да креирамо документ и тестирамо је. Такође можемо да погледамо у обимну ConTeXt документацију. Као што је уобичајено у свету слободног софтвера, „ConTeXt Standalone” садржи изворне фајлове скоро комплетне документације у дистрибуцији. Када желимо да знамо да ли се опција која нас интересује употребљава у било ком од ових изворних фајлова, алат као што је „*grep*” (за GNU Linux системе) нам може помоћи да их све претражимо и видимо на конкретном примеру како се опција користи и шта ради.

Ово је начин на који је замишљено учење система ConTeXt: увод детаљно објашњава пет (тачније четири) аспекта које сам истакао, и још више: док читамо, у глави ћемо формирати јасну слику редоследа: *команда која извршава задатак – команда која конфигурише претходну – команда која омогућава креирање сличне команде*. Такође ћемо научити и неке од основних структура система ConTeXt и знаћемо за шта се користе.

Глава 4

Изворни фајлови и пројекти

Садржај: 4.1 Кодирање изворних фајлова; 4.2 Карактери у изворном фајлу које ConTeXt третира на посебан начин; 4.2.1 Размаци и табови; 4.2.2 Преломи линија; 4.2.3 Линије/црте; 4.3 Прости пројекти и пројекти са више фајлова; 4.4 Структура изворног фајла у простим пројектима; 4.5 Рад са више фајлова у T_EX стилу; 4.5.1 Команда \input; 4.5.2 \ReadFile и \readfile; 4.6 ConTeXt пројекти у ужем смислу; 4.6.1 Фајлови окружења; 4.6.2 Компоненте и производи; 4.6.3 Пројекти у ужем смислу; 4.6.4 Заједнички аспекти окружења, компоненти, производа и пројеката;

Као што већ знамо, када радимо у систему ConTeXt увек почињемо текст фајлом у којем се, уз садржај финалног документа, налази већи број инструкција које наводе трансформације помоћу којих ConTeXt из изворног фајла генерише коректно форматирани излазни документ у PDF формату.

Имајући у виду читаоце који су до сада познавали само рад у текст процесорима, мислим да вреди уложити нешто времена у сам изворни фајл. Или пре, изворне фајлове, јер понекада постоји само један изворни фајл, а некада користимо већи број изворних фајлова да дођемо до финалног документа. У овом последњем случају говоримо о „пројектима са више фајлова”.

4.1 Кодирање изворних фајлова

Изворни фајл(ови) мора(ју) бити текст фајл(ови). У компјутерској терминологији, ово је име које се даје фајлу који садржи само читљиви текст и који не садржи никакав бинарни код. Ови фајлови се такође називају и *прости текст* или *чисти текст* фајлови.

Пошто компјутерски системи интерно обрађују само бинарне бројеве, текст фајл се уствари састоји од *бројева* који представљају *карактере*. За повезивање броја са карактером се користи *табела*. За текст фајлове постоји више могућих табела. Израз *кодирање текст фајла* се односи на одређену табелу мапирања карактера коју користи дати текст фајл.

Постојање различитих табела кодирања текст фајлова је последица само историје компјутерске науке. У раним фазама развоја, када је капацитет меморије и складишта компјутерских уређаја био скроман, одлучено је да се користи табела под именом ASCII (што је акроним од „*American Standard Code for Information Interchange*”) која је дозвољавала само 128 карактера и успоставио ју је Амерички комитет за стандарде 1963. године. Очигледно је да 128 карактера није довољно да се представе сви карактери и симболи који се користе у свим светским језицима; али је било више него довољно да се представи енглески језик који, од свих западних језика има најмање карактера, јер не употребљава дијакритике (акценте и остале знаке изнад, испод, или кроз остала слова). Предност употребе ASCII табеле је што текст фајлови заузимају врло мало простора, јер 127 (највећи број у табели) може да се представи са бинарним бројем од 7 цифара, а први компјутери су користили бајт за мерење меморије, односно бинарни број са 8 цифара. Било који карактер у табели је могао да се смести у један бајт. Пошто бајт има 8 цифара, а ASCII користи само 7, чак је остало простора да се додају у неки други карактери којима могу да се представе остали језици.

Али када се употреба компјутера проширила, постало је очигледно да ASCII није адекватан, па је дошло до развоја *алтернативних* табела у којима су се налазили карактери ван енглеског алфабета, као што је шпанско 'ñ', акцентовани самогласници, каталонско или француско 'ç', итд. С друге стране, није било почетног договора како би те ASCII *алтернативне* табеле требало да изгледају, тако да су различите специјализоване компјутерске компаније постепено саме решавале проблем. Из тог разлога, не само да су развијене одрешене табеле за различите језике или групе језика, већ и различите табеле према компанији која их је креирала (Microsoft, Apple, IBM, итд.).

Идеја да се креира једна табела која би обухватила све језике се појавила тек са повећањем компјутерске меморије и смањеном цене уређаја за складиштење. Али да поновимо, уствари се није креирала једна табела која садржи све карактере,

већ стандардно кодирање (под називом Уникод) уз различите начине за његово представљање (UTF-8, UTF-16, UTF-32, итд.) Од свих ових система, UTF-8 је на крају постао де факто стандард којим може да се представи практично било који живи језик, као и многи већ изумрли језици и бројни додатни симболи. Сви они користе бројеве променљиве дужине (између 1 и 4 бајтова), што у суштини помаже да се оптимизује величина текст фајлова. Ова величина се није значајно увећала у односу на текст фајлове који користе чисти ASCII.

Све док се није појавио ХџџХ, системи засновани на ТџХ – који су такође рођени у САД, па су зато користили енглески као свој природни језик – претпостављали су да је кодирање чисти ASCII; а ако сте желели да користите неко друго кодирање, морали сте на неки начин да га наведете у изворном фајлу.

ConTeXt Mark IV претпоставља да је кодирање UTF-8. Ипак, на мало старијим компјутерским системима се можда још увек као подразумевано кодирање користи неко друго. Нисам потпуно сигуран које подразумевано кодирање користи Windows, када се има у виду стратегија компаније Microsoft да скривањем сложености дође до шире публике (али мада је скривена, то не значи да је елиминисана!). Нема много доступних информација (или можда ја нисам успео да их пронађем) у вези система кодирања који подразумевано користи.

У сваком случају, које год да је подразумевано кодирање, сваки текст едитор вам омогућава да фајл сачувате у жељеном кодирању. Изворни фајлови које треба да обради ConTeXt Mark IV морају да се сачувају у UTF-8, осим, наравно, када постоји веома добар разлог да се употреби неко друго (мада не могу да смислим ниједан валидан разлог за то).

Ако желимо да пишемо фајл у неком другом кодирању (можда неки стари фајл) можемо да

- Конвертујемо фајл у UTF-8, што се препоручује, и за то постоје разни алати; на Linux систему, на пример, команде `iconv` или `recode`.
- Наведемо у изворном фајлу да кодирање није UTF-8. Да бисмо то урадили, морамо да употребимо команду `\enableregime`, чија је синтакса:

`\enableregime[Кодирање]`

где се *Кодирање* односи на име под којим систем ConTeXt зна које је стварно кодирање фајла у питању. У [табели 4.1](#) ћете пронаћи различита кодирања и имена под којим их зна систем ConTeXt.

| Кодирање | Име(на) у ConTeXt | Напомене |
|--------------------------|---|-----------------|
| Windows CP 1250 | cp1250, windows-1250 | Западноевропско |
| Windows CP 1251 | cp1251, windows-1251 | Ћирилично |
| Windows CP 1252 | cp1252, win, windows-1252 | Западноевропско |
| Windows CP 1253 | cp1253, windows-1253 | Грчко |
| Windows CP 1254 | cp1254, windows-1254 | Турско |
| Windows CP 1257 | cp1257, windows-1257 | Балтичко |
| ISO-8859-1, ISO Latin 1 | iso-8859-1, latin1, il1 | Западноевропско |
| ISO-8859-2, ISO Latin 2 | iso-8859-2, latin2, il2 | Западноевропско |
| ISO-8859-15, ISO Latin 9 | iso-8859-15, latin9, il9 | Западноевропско |
| ISO-8859-7 | iso-8859-7, grk | Грчко |
| Mac Roman | mac | Западноевропско |
| IBM PC DOS | ibm | Западноевропско |
| UTF-8 | utf | Уникод |
| VISCII | vis, viscii | Вијетнамско |
| DOS CP 866 | cp866, cp866nav | Ћирилично |
| KOI8 | koi8-r, koi8-u, koi8-ru | Ћирилично |
| Mac Cyrillic | maccyr, macukr | Ћирилично |
| Остала | cp855, cp866av, cp866mav, cp866tat, ctt, dbk, iso88595, isoir111, mik, mls, mnk, mos, ncc | Разна |

Табела 4.1 Главна кодирања у систему ConTeXt

ConTeXt Mk IV снажно препоручује употребу UTF-8. И ја се слажем са овом препоруком. Од сада па надаље кроз овај увод, можемо претпоставити да је кодирање увек UTF-8.



Уз команду `\enableregime` ConTeXt има и команду `\useregime` која нам омогућава да као њен аргумент наведемо код за једно или остала кодирања. Нисам пронашао никакве информације о овој команди, нити како се она разликује од `\enableregime`, већ само неке примере употребе. Претпостављам да је `\useregime` дизајнирана за сложене пројекте који користе много изворних фајлова, уз очекивање да немају сви исто кодирање. Али то је само нагађање.

4.2 Карактери у изворном фајлу које ConTeXt третира на посебан начин

Специјални карактери је име којим ћу називати групу карактера који се разликују од *резервисаних карактера*. Као што смо видели у одељку 3.1, то су они који за систем ConTeXt имају посебно значење, тако да се не могу директно употребљавати као карактери у изворном фајлу. Уз њих, постоји још једна група карактера која, мада их ConTeXt третира као такве када у изворном фајлу наиђе на њих, они се не третирају посебним правилима. У овој групи се налазе размаци, табови, преломи линија и цртице.

4.2.1 Размаци и табови

Табови и размаци се у изворном фајлу за све сврхе третирају на исти начин. ConTeXt ће таб карактер (Tab тастер на тастатури) трансформисати у празан простор. А размаци се апсорбују у било који други празан простор (или таб) који се налази непосредно иза њих. Тако да нема апсолутно никакве разлике ако у изворном фајлу напишете

Станлио и Олио.

или

Станлио и Олио.

ConTeXt сматра да су ове све реченице потпуно исте. Дакле, ако између речи желимо да унесемо додатни размак, морамо да употребимо неке ConTeXt команде које то раде. Обично ће радити са „\ ”, што значи карактер \ иза којег се налази размак. Али постоје и остале процедуре у вези хоризонталног размака које ћемо представити у поглављу 10.3.

Апсорпција узастопних размака нам омогућава да изворни фајл пишемо тако да визуелно истакнемо делове које желимо да нагласимо, једноставно повећавајући или умањујући коришћено увлачење, и да не бринемо јер знамо да то неће уопште утицати на финални документ. Дакле, у следећем примеру

```
Музичка група из Мадрида с краја седмдесетих {\em La Romántica
Banda Local} је писала песме еклетичког стила које је било врло тешко
ставити у неку категорију. На пример, у својој песми „El Egiptio” су рекли:
\quotation{{\em Esto es una farsa más que una comedia, página muy seria
de la histeria musical; sueños de princesa, vicios de gitano pueden en
su mano acariciar la verdad}}, мешајући речи, фразе, просто зато што
поседују унутрашњи ритам (comedia-histeria-seria, gitano-mano).
```

можемо видети да су неке линије мало увучене удесно. То су линије које су део фрагмената који би требало да се појаве у курзиву. Постојање ових увлачења помаже (аутору) да види где се курзив завршава.

Неко би помислио, каква збрка! Морам ли да се замарам увлачењем линија? Истина је да ово посебно увлачење аутоматски ради мој едитор (GNU Emacs) када уређујем ConTeXt изворни фајл. То је та врста мале помоћи услед које бирате да радите са неким едитором, а не са неким другим.

Правило апсорпције размака се примењује само на узастопне размаке у изворном фајлу. Дакле, ако се у изворни фајл између два размака постави празна група („{ }”), мада она у финалном фајлу неће произвести ништа, њено присуство ће обезбедити да два размака нису узастопна. На пример, ако напишемо „Станлио { } и Олио”, добићемо „Станлио и Олио”, где, ако пажљиво погледате, између прве две речи постоје два узастопна размака.

Исто се дешава и са резервисаним карактером „~”, мада је његов ефекат да генерише размак чак и ако то није: размак иза кога дође ~ неће апсорбовати наредне, а неће ни размак након ~.

4.2.2 Преломи линија

Када линија достигне максималну ширину, у већини едитора се аутоматски уноси прелом линије. Прелом линије можемо такође и експлицитно да унесемо тако што притиснемо тастер „Ентер” или „Return”.

ConTeXt примењује следећа правила у вези прелома линије:

- a. За све намене, један прелом линије је исто што и размак. Дакле, ако се непосредно испред или иза прелома линије нађе размак или таб, прелом линије или први размак ће их апсорбовати, а у финални документ ће се уметнути прости размак.
- b. Два или више узастопна прелома линије креирају прелом пасуса. У овом смислу, сматра се да су два прелома линије узастопна ако се између првог и другог не налази ништа друго осим размака или табова (јер их први прелом линије апсорбује); што укратко значи да једна или више узастопних линија које су у изворном фајлу потпуно празне (без икаквих карактера или само са размацима или табовима) постају прелом пасуса.

Запазите да сам рекао „два или више узастопна прелома линије”, па затим „једна или више празних узастопних линија”, што значи да ако желимо да повећамо размак између пасуса, то не радимо једноставним уметањем још једног прелома линије. За то морамо да употребимо команду која повећава вертикални размак. Ако нам треба само једна додатна линија раздвајања, можемо да употребимо команду `\blank`. Али постоје и остале процедуре за повећање вертикалног размака. Указујем на одељак 11.2.

У неким приликама, када прелом линије постаје размак, може да се јави неки нежељени и неочекивани размак. Посебно када пишемо макрое, јер је тада лако да се размак „ушуња” а да то не приметимо. Да бисмо то спречили, можемо употребити резервисани карактер „%” који, као што знамо, тамо где се појави спречава обраду линије, што повлачи да се не обрађује ни прелом на крају реда. Тако на пример, команда

```
\define[3]\Test{
  {\em #1}
  {\bf #2}
  {\sc #3}
}
```

која свој први аргумент испишује у курзиву, други у црном слогу, а трећи капиталом би уметнула размак између сваког од ових аргумената, док

```
\define[3]\Test{%
  {\em #1}%
  {\bf #2}%
  {\sc #3}%
}
```

неће уметнути никакав размак између њих, јер резервисани карактер % спречава да се прелом линије обради и он постаје само размак.

4.2.3 Линије/црте

Црте су добар пример разлике између компјутерске тастатуре и штампаног текста. На нормалној тастатури, обично постоји само један карактер за црту (или линију у типографском изразу) који зовемо цртица или („-”); али штампани текст користи до четири различите дужине линија:

- Кратке линије (цртице), као оне које служе за раздвајање речи на крају реда (-).
- Линије средње дужине (ен црте или ен линије), мало дуже од претходних (-). Имају већи број примена, у неким европским језицима (не толико у енглеском) означавају почетак линије дијалога, или раздвајају мањи од већег броја опсега у датумима или страницама; „стр. 12–33”.
- Дугачке линије (ем црте или ем линије) (—), користе се уместо заграда, за укључивање једне реченице унутар друге.
- Минус знак (–) за представљање одузимања или негативног броја.

Данас је у UTF-8 кодирању доступно све горе наведено и још више. Али пошто још увек не могу све да се генеришу притиском на један тастер на тастатури, није тако једноставно да се уметну у изворни фајл. \TeX је на срећу увидео потребу да се у финални документ умеће више линија/црта него што може да се произведе тастатуром, па је дизајнирао просту процедуру за уметање. $\text{Con}\TeX$ је ту процедуру проширио додавањем команди које генеришу разне врсте линија. За генерисање четири врсте линија можемо употребити два приступа: било обичан $\text{Con}\TeX$ начин употребом команде, или директно са тастатуре. Ове процедуре су приказане у [табели 4.2](#):

| Врста линије | Изглед | Написана директно | Команда |
|--------------|--------|-------------------|----------------------|
| Цртица | - | - | <code>\hyphen</code> |
| Ен линија | - | -- | <code>\endash</code> |
| Ем линија | — | --- | <code>\emdash</code> |
| Знак минус | - | \$-\$ | <code>\minus</code> |

Табела 4.2 Линије/црте у $\text{Con}\TeX$

Имена команди `\hyphen` и `\minus` су она која са обично користе у енглеском језику. Мада их многи у штампарској индустрији називају 'линије', \TeX појмови, тачније `\endash` и `\emdash` су такође уобичајени у словослагачкој терминологији. „ен” и „ем” су имена мерних јединица које се користе у типографији. „Ен” представља ширину карактера 'n', док „ем” представља ширину карактера 'm' у фонту који се користи.

4.3 Прости пројекти и пројекти са више фајлова

У систему $\text{Con}\TeX$ можемо да користимо само један изворни фајл којем се налази комплетан садржај финалног документа као и сви детаљи везани за њега, па у том случају говоримо о „простим пројектима”, или супротно од тога, могли бисмо да употребимо више изворних фајлова који деле садржај финалног документа, па у том случају говоримо о „пројектима са више фајлова”.

Ситуације у којима је уобичајено да се ради са више изворних фајлова су следеће:

- Ако пишемо документ на којем сарађује више аутора, од којих сваки пише по један део на којем ради само он; на пример, ако пишемо фестшрифт са прилозима различитих аутора, или број неког журнала, итд.
- Ако је документ на којем радимо, просто речено велики, тако да се компјутер успори када га уређујемо или када га компајлирамо; у том случају, подела материјала на неколико изворних фајлова у знатној мери убрзава компајлирање сваког дела.
- Исто тако, ако смо написали више макроа које желимо да применимо у неким (или свим) нашим документима, или ако смо генерисали шаблон који контролише или стилизује наше документе и хоћемо да га применимо на сваки документ, итд.

4.4 Структура изворног фајла у простим пројектима

Структура простих пројеката који се развијају само у једном изворном фајлу је веома једноставна и заснива се око „text” окружења које, у суштини, мора да се појави у том истом фајлу. Разликујемо следеће делове фајла:

- **Преамбула документа:** све од прве линије фајла до почетка „text” окружења (`\starttext`).
- **Тело документа:** ово је садржај „text” окружења; или другим речима, све што се налази између `\starttext` и `\stoptext`.

```

% Прва линија документа

% Област преамбуле:
% Садржи глобалне конфигурационе команде
% документа

\starttext % Овде почиње тело документа

...
... % Садржај документа
...

\stoptext % Крај документа

```

Слика 4.1 фајл који садржи прости пројекат

На [слици 4.1](#) видимо веома прост изворни фајл. Потпуно све испред команде `\starttext` (која је на слици у линији 5, ако се броје само линије са текстом), чини преамбулу; све између `\starttext` и `\stoptext` чини тело документа. Било шта након `stoptext` се игнорише.

Преамбула се користи за уметање команди које треба да утичу на документ као целину, односно оне које одређују његову глобалну конфигурацију. Уопште није обавезно да се у преамбули напише било каква команда. Ако ту нема команди, ConTeXt ће усвојити подразумевану конфигурацију која није баш детаљно развијена, али ће послужити за многе документе. У добро планираним документима, преамбула ће садржати све команде које утичу на документ као на целину, као што су макрои и прилагођене команде које ће се користити у изворном фајлу. У типичној преамбули, то би могло да изгледа овако:

- Назначавање главног језика документа (погледајте [одељак 10.5](#)).
- Назначавање величине папира ([одељак 5.1](#)) и распореда стране ([одељак 5.3](#)).
- Особине главног фонта документа ([одељак 6.3](#)).
- Прилагођавања section команди које ће се користити ([одељак 7.4](#)) и, ако је потребно, дефиницију нових section команди ([section 7.5](#)).
- Распоред заглавља и подножја ([одељак 5.6](#)).
- Подешавања за наше нове сопствене макрое ([section 3.7](#)).
- Итд.

Преамбула је намењена за глобалну конфигурацију документа; тако да ту не би требало да буде ништа што се тиче *садржаја* документа, или текста који се обрађује. У теорији, сав текст за обраду се у преамбули игнорише, мада понекад, ако је ту, изазваће грешку компајлирања.

Тело документа, окружено командама `\starttext` и `\stoptext` садржи стварни садржај, што значи текст који се обрађује, заједно са ConTeXt командама које не би требало да утичу на цео документ.

4.5 Рад са више фајлова у T_EX стилу

Да би омогућио рад са више изворних фајлова, T_EX је понудио примитиву која се зове `\input`. Та примитива такође функционише и у систему ConTeXt, мада он прихвата и две специфичне команде које донекле усавршавају начин на који функционише `\input`.

4.5.1 Команда `\input`

Команда `\input` умеће садржај фајла који јој се проследи као аргумент. Њен формат је:

```
\input ИмеФајла
```

где је *ИмеФајла* име фајла који треба да се уметне. Обратите пажњу да име не мора да се постави унутар витчастих заграда, мада се неће јавити грешка чак и ако се то уради. Међутим, име никада не би требало поставити у велике заграде. Ако је екстензија фајл „.tex“, може да се изостави.

Када ConTeXt компајлира документ и наиђе на `\input` команду, он тражи наведени фајл и наставља компајлирање као да је тај фајл био део фајла који га читава. Када заврши компајлирање, он се враћа у оригинални фајл и наставља са места одакле је ушао у читавани фајл; тако да је резултат, практично, као да је садржај фајла наведеног у команди `\input` уметнут на место на којем се она налази. Фајл који се позива командом `\input` мора имати важеће име у нашем оперативном систему и не сме да садржи размаке. ConTeXt ће га потражити у радном директоријуму, па ако га ту не пронађе, потражиће га у директоријумима наведеним у TEXROOT променљивој окружења. Ако се фајл на крају не пронађе, вратиће се грешка компајлирања.

Најчешћа употреба `\input` команде изгледа овако: запише се фајл, назовимо га „`glavni.tex`“, који ће се користити као контејнер за позивање `\input` командом разних фајлова који чине наш пројекат. Ово је приказано у следећем примеру:

```
% Опште конфигурационе команде:

\input MojaKonfiguracija

\starttext

\input NaslovnaStrana
\input Predgovor
\input Glava1
\input Glava2
\input Glava3

...

\stoptext
```

Запазите како смо за општу конфигурацију документа позвали фајл „`МојаКонфигурација.tex`“ за који се претпоставља да садржи глобалне команде које желимо да применимо. Затим, између команди `\starttext` и `\stoptext` позивамо неколико фајлова у којима се налази садржај различитих делова нашег документа. Ако у неком тренутку пожелимо да убрзамо процес компајлирања, потребно је да изоставимо компајлирање неких фајлова, па је потребно само да на почетак линије која позива одређени фајл ставимо знак коментара. На пример, ако пишемо треће поглавље и желимо да га компајлирамо једноставно само да проверимо има ли у њему грешака, не морамо да компајлирамо остатак, па можемо да напишемо:

```
% Опште конфигурационе команде:

\input MojaKonfiguracija

\starttext

% \input NaslovnaStrana
% \input Predgovor
% \input Glava1
% \input Glava2

\input Glava3

...

\stoptext
```

па ће се компајлирати само Глава 3. С друге стране, запазите да измена редоследа поглавља значи једноставно измену редоследа линија које их позивају.

Када у пројекту који се састоји из више фајлова изузмемо неки фајл из процеса компајлирања, добијамо на брзини обраде, али резултат тога је да све референце из дела који се не компајлира а указују на остале делове који још увек нису компајлирани неће више радити. Погледајте [одељак 9.2](#).

Важно је да буде јасно да када радимо са командом `\input`, само главни фајл, онај из којег се позивају сви остали, сме да садржи команде `\starttext` и `\stoptext`, јер ако се налазе и у осталим фајловима,

јавиће се грешка компајлирања. С друге стране, ово значи да не можемо директно да компајлирамо појединачне фајлове који чине пројекат, већ је неопходно да се они компајлирају из главног фајла, то јест оног који наводи основну структуру документа.

4.5.2 `\ReadFile` и `\readfile`

Као што смо управо видели, ако `ConTeXt` не пронађе фајл који се позива командом `\input`, јавиће се грешка. У ситуацији када желимо да уметнемо фајл само ако постоји, а дозвољава се и могућност да фајл можда не постоји, `ConTeXt` пружа варијацију команде `\input`. То је

`\ReadFile{ИмеФајла}`

Ова команда је слична команди `\input` у сваком погледу, осим што у случају да се фајл не пронађе, компајлирање се наставља и не јавља се никаква грешка. Такође се разликује од команде `\input` и по синтакси, јер знамо да за `\input` није обавезно да се име фајла постави у витичасте заграде. Али са `\ReadFile` је то обавезно. Ако не употребимо витичасте заграде, `ConTeXt` ће мислити да је име фајла који треба да нађе исто као и први карактер који следи иза команде `\ReadFile`, уз екстензију `.tex`. Тако да ако, на пример, напишемо

`\ReadFile MojFajl`

`ConTeXt` ће разумети да фајл који треба да прочита има име „`M.tex`”, јер је карактер непосредно након команде `\ReadFile` (изузимајући размаке који се, као што знамо, игноришу на крају имена команде) `'M'`. Пошто `ConTeXt` у општем случају неће пронаћи фајл под именом „`M.tex`”, а `\ReadFile` не генерише грешку ако не пронађе фајл, `ConTeXt` ће наставити компајлирање након `'M'` у „`MojFajl`”, па ће да уметне текст „`ojFajl`”.

`\readfile` је софистициранија верзија команде `\ReadFile` чији је формат

`\readfile{ИмеФајла}{ТекстАкоПостоји}{ТекстАкоНеПостоји}`

Први аргумент је сличан аргументу за `\ReadFile`: то је име фајл унутар витичастих заграда. Други аргумент је текст који се исписује у случају да фајл постоји, пре него него што се садржај фајла уметне. Трећи аргумент је текст који се исписује ако се наведени фајл не пронађе. То значи да у зависности од тога да ли се пронађе фајл наведен као први аргумент, извршавају се други (ако фајл постоји), или трећи аргумент (ако фајл не постоји).

4.6 `ConTeXt` пројекти у ужем смислу

Трећи механизам који систем `ConTeXt` нуди за рад на пројектима са више фајлова је сложенији и комплетнији: он почиње тако што се прави разлика између фајлова пројекта, фајлова производа, фајлова компоненти и фајлова окружења. Да бисмо разумели међусобне везе и функцију сваког од ових типова, сматрам да је најбоље објаснити сваки од њих појединачно:

4.6.1 Фајлови окружења

Фајл окружења је фајл који чува макрое и конфигурације специфичних стилова намењених да се примене на неколико докумената, било да су они потпуно независни документи, било да су делови неког сложеног документа. Дакле, фајл окружења може да садржи све што бисмо иначе писали испред `\starttext`; то јест: општу конфигурацију документа.

За ове врсте фајлова сам задржао израз „фајлови окружења” како се не бих удаљио од званичне `ConTeXt` терминологије, мада верујем да би погоднији израз вероватно био „фајлови формата” или „фајлови глобалне конфигурације”.

Као и сви `ConTeXt` изворни фајлови, фајлови окружења су текст фајлови који подразумевају да је екстензија „`.tex`”, мада ако то желимо, можемо да је променимо, највероватније на „`.env`”. Међутим, у систему `ConTeXt` се ово обично не ради. Најчешће се фајл окружења препознаје тако што почиње

или се завршава са 'env'. На пример: „`MojiMakroi_env.tex`” или „`env_MojiMakroi.tex`”. Унутрашњост једног таквог фајла окружења би могла да изгледа овако:

```
\startenvironment MojeOkruzenje

\mainlanguage[sr]

\setupbodyfont
[dejavu]

\setupwhitespace
[big]

...

\stopenvironment
```

Или исказано на други начин, дефиниције и конфигурационе команде се постављају између `\startenvironment` и `\stopenvironment`. Непосредно иза `\startenvironment` пишемо име којим називамо то окружење, па онда наводимо све команде које треба да чине окружење.

Што се тиче имена окружења, према резултатима мојих тестова, име које стављамо непосредно иза `\startenvironment` је чисто индикативно, тако да и ако га не наведемо, ништа (лоше) се не догађа.

Предвиђено је да фајлови окружења функционишу заједно са компонентама и производима (који су објашњени у наредном одељку). То је разлог због којег једно или више окружења може да се позове из компоненте или производа командом `\environment`. Али ова команда такође ради и ако се употреби у конфигурационој области (преамбули) било ког ConT_EXt изворног фајла, чак и ако то није изворни фајл предвиђен да се компајлира у деловима.

Команда `\environment` може да се позове употребом било којег од следећа два формата:

`\environment` Фајл

`\environment[Фајл]`

У сваком случају, резултат команде ће бити учитавање садржаја фајла који се наведе као њен аргумент. Ако се тај фајл не пронађе, компајлирање ће се наставити на уобичајен начин, без враћања било какве грешке. Ако је екстензија фајла „`.tex`”, може да се изостави.

4.6.2 Компоненте и производи

Ако замислимо књигу у којој је свако поглавље у различитом изворном фајлу, онда можемо да кажемо да су поглавља *компоненте*, а да је књига *производ*. Ово значи да је *компонента* самостални део *производа*, који може да поседује сопствени стил и може да се компајлира независно. Свака компонента ће имати различит фајл, а уз то ће постојати и фајл производа који обједињује све компоненте у целину.

Типични фајл компоненте изгледа овако

```
\environment MojeOkruzenje
\environment MojiMakroi

\startcomponent Glava1

\startchapter[title={Глава 1}]

...

\stopcomponent
```

А фајл производа би могао да изгледа овако:

```
\environment MojeOkruzenje
\environment MojiMakroi

\startproduct MojaKnjiga

  \component Glava1
  \component Glava2
  \component Glava3

  ...

\stopproduct
```

Запазите да ће стварни садржај нашег документа бити распоређен по разним фајловима ‘компоненти’, а да је фајл производа ограничен на успостављање редоследа компоненти. С друге стране, (појединачне) компоненте и производи могу директно да се компајлирају. Компајлирање производа ће генерисати PDF фајл који садржи све компоненте тог производа. А ако се компајлира појединачна компонента, то ће генерисати PDF фајл који садржи само компоненту која се компајлира.

Унутар фајла компоненте, пре команде `\startcomponent`, са `\environment ИмеОкружења` можемо да позовемо један или више фајлова окружења. Исто можемо да урадимо и у фајлу производа, пре `\startproduct`. Истовремено може да се учита неколико фајлова окружења. На пример, можемо да имамо своју омиљену колекцију макроа и разних стилова које примењујемо на документе који се налазе у различитим фајловима. Међутим, имајте на уму да када користимо два или више окружења, она се читавају у редоследу у којем су позивана, тако да ако се иста конфигурациона команда налазу у више од једног окружења, а има различите вредности, примениће се вредности оног последње учитаног. С друге стране, окружења се читавају само једном, тако да ако у претходним примерима у којима се окружење позива из фајла производа и из одређених фајлова компоненти, ако компајлирамо производ, то је тренутак читавања окружења у редоследу у којем су ту наведена; када се окружење позове из било које од компоненти, ConTeXt ће проверити да ли је окружење већ једном било учитано, па у том случају неће урадити ништа.

Име компоненте која се позива из производа мора бити име фајла који садржи ту компоненту, мада ако је екстензија тог фајла „.tex”, она може да се изостави.

4.6.3 Пројекти у ужем смислу

У већини случајева је разлика између производа и компоненти довољна. Исто тако, ConTeXt поседује чак и виши ниво у којем можемо да групишемо већи број производа: то је *пројекат*.

Типични фајл пројекта би отприлике изгледао овако

```
\startproject MojaKolekcija

  \environment MojeOkruzenje
  \environment MojiMakroi

  \product Knjiga1
  \product Knjiga2
  \product Knjiga3

  ...

\stopproject
```

Сценарио у којем би нам био потребан пројекат је, на пример, онај у којем је потребно да уредимо колекцију књига, све са истим спецификацијама формата; или ако уређујемо неки журнал: таква колекција књига или журнала би била пројекат; свака књига или сваки број журнала би био производ; а свако поглавље књиге или сваки чланак у броју журнала би био компонента.

С друге стране, пројекти нису предвиђени да се директно компајлирају. Имајте на уму да би по дефиницији сваки производ који припада пројекту (свака књига у колекцији, или сваки број журнала)

требало да се компајлира одвојено и да се генерише његов PDF. Стога се у њега поставља команда `\product` којом се назначава који производи припадају неком пројекту, а која у суштини не ради ништа: она је просто подсетник за аутора.

Јасно је да се неко може запитати зашто имамо пројекте ако не могу да се компајлирају: одговор је зато што фајл пројекта везује одређена окружења за пројекат. То је разлог што ће ConTeXt, ако у фајл компоненте или производа уметнемо команду `\project ИмеПројекта`, да прочита фајл пројекта и аутоматски да учита окружења везана за њега. Зато у пројектима команда `\environment` мора да дође након `\startproject`; међутим, у производима и компонентама, `\environment` мора да дође *пре* `\startproduct` или `\startcomponent`.

Исто као и са командама `\environment` и `\component`, команда `\project` нам омогућава да наведемо име пројекта било унутар великих заграда, било да уопште не користимо велике заграде. Ово значи да су `\project ИмеФајла` и `\Project[ИмеФајла]` еквивалентне команде.

Резиме различитих начина за учитавање окружења

Из претходног следи да окружење може да се учита било којом од следећих процедура:

- а. Уметањем команде `\environment ИмеОкружења` испред `\starttext` или `\startcomponent`. То ће да учита окружење само за компајлирање тог фајла.
- б. Уметањем команде `\environment ИмеОкружења` у фајл производа испред `\startproduct`. То ће да учита окружење када се компајлира производ, али неће у случају да се његове компоненте компајлирају посебно.
- в. Уметањем команде `\project` у производ или окружење: то ће да учита сва окружења везана за пројекат (у фајлу пројекта).

4.6.4 Заједнички аспекти окружења, компоненти, производа и пројекта

Имена окружења, компоненти, производа и пројекта: Већ смо видели да се, за све ове елементе, након `\start` команде која започне одређено окружење, компоненту или производ, његово име мора ручно да се наведе. Као правило, ово име мора да се подудара са именом фајла који садржи окружење, компоненту ли производ јер, на пример, када ConTeXt компајлира производ, па сагласно са фајлом производа мора да учита окружење или компоненту, не постоји начин да се зна који фајл представља то окружење или компоненту осим ако фајл има исто име као и елемент који треба да се учита.

Иначе је, према резултатима мојих тестова, име које се пише након `\startproduct` или `\startenvironment` у фајловима производа и окружења чисто индикативно. Ако се изостави, или ако се не подудара са именом фајла, не дешава се ништа лоше. Међутим, у случају компоненти је важно да се име компоненте подудара са именом фајла који је садржи.

Структура директоријума пројекта: Знамо да ConTeXt подразумевано тражи фајлове у радном директоријуму и на путањи коју наводи променљива TEXROOT. Међутим, када користимо `\project`, `\product`, `\component` или `\environment` команде, претпоставља се да пројекат има структуру директоријума у којој се заједнички елементи налазе у директоријуму родитељу, а они специфични у неким дете директоријумима. Дакле, ако се наведени фајл не пронађе у радном директоријуму, потражиће се у његовим родитељ директоријумима, па ако се ни тамо не пронађе, у родитељ директоријуму тог директоријума и тако даље.

II

Глобални аспекти документа

Глава 5

Странице и пагинација документа

Садржај: **5.1 Величина странице;** 5.1.1 Подешавање величине странице; 5.1.2 Употреба нестандартних величина странице; 5.1.3 Промена величине папира на било ком месту у документу; 5.1.4 Подешавање величине странице према садржају; **5.2 Елементи на страници;** **5.3 Распоред странице (\setuplayout);** 5.3.1 Додела величине различитим компонентама странице; 5.3.2 Подешавање распореда странице; 5.3.3 Употреба више различитих распореда страница; 5.3.4 Остале ствари у вези распореда странице; А Разликовање непарних и парних страница; Б Странице са више од једне колоне; **5.4 Нумерација страница;** **5.5 Форсирани или предложени преломи странице;** 5.5.1 Команда \page; 5.5.2 Спајање одређених линија пасуса тако да се спречи уметање прелома странице између њих; **5.6 Заглавља и подножја;** 5.6.1 Команде за постављање саржаја заглавља и подножја; 5.6.2 Форматирање заглавља и подножја; 5.6.3 Дефинисање специфичних заглавља и подножја и везивање за section команде; **5.7 Уметање текст елемената у ивице и маргине странице;**

ConTeXt трансформише изворни документ у коректно форматиране *странице*. Изглед страница, распоред текста и празнина, као и елементи на страницама су од пресудног значаја за словослагање. Ово поглавље је посвећено свим овим питањима, и још неким стварима које се тичу пагинације.

5.1 Величина странице

5.1.1 Подешавање величине странице

ConTeXt подразумевано претпоставља да ће документ бити A4 величине, односно следи европски стандард. Командом `\setuppapersize` можемо да подесимо неку другу величину. То је команда која се типично поставља у преамбулу документа. *Уобичајена* синтакса ове команде је:

`\setuppapersize[ЛогичкаСтраница][ФизичкаСтраница]`

где су оба аргумента симболичка имена.¹ Први аргумент који сам назвао *ЛогичкаСтраница*, представља величину странице која треба да се узме у обзир за словослагање; а други аргумент *ФизичкаСтраница*, представља величину папира на којој ће се документ штампати. Обично су обе величине исте, па онда други аргумент може да се изостави; међутим, у ситуацијама када су две величине различите, на пример, када се књига штампа у листовима од по 8 или 16 страница (уобичајена техника штампе, посебно за академске књиге до око 1960их година). У тим случајевима ConTeXt нам омогућава да направимо разлику између обе величине; а ако је идеја да се неколико страница штампа на једном листу папира, можемо такође да наведемо и шему савијања која треба да се поштује користећи команду `\setuparranging` (за коју се у овом уводу не даје објашњење).

За величину словослагања можемо да наведемо било коју од предефинисаних величина које се користе у индустрији папира (или коју ми користимо). То су:

- Било која из А, В и С серија дефинисаних ISO-216 стандардом (од A0 до A10, од B0 до B10 и од C0 до C10), обично се користе у Европи.
- Било која од америчких стандардних величина: letter, ledger, tabloid, legal, folio, executive.

¹ Присетимо се да сам у одељку 3.5 навео да у општем случају постоје две врсте опција које примају ConTeXt команде: симболичка имена, чије значење ConTeXt већ познаје, или променљиве чије вредности морамо експлицитно да наведемо.

- Било која од RA и RSA величина дефинисаних ISO-217 стандарду.
- G5 и E5 величина дефинисаних швајцарским SIS-014711 стандардом (за докторске тезе).
- За коверте: било која од величина дефинисаних северноамеричким стандардом (envelope 9 до envelope 14) или ISO-269 стандардом (C6/C5, DL, E4).
- CD, за CD омоте.
- S3 – S6, S8, SM, SW за величине екрана у документима који нису намењени штампању, већ приказу на екран.

Заједно са величином папира, командом `\setuppapersize` можемо да назначимо и оријентацију странице: „portrait” (вертикална) или „landscape” (хоризонтална).



Према ConTeXt викију, остале опције које прихвата `\setuppapersize` су „rotated”, „90”, „180”, „270”, „mirrored” и „negative”. У мојим личним текстовима сам приметио неке видљиве промене са „rotated” које инвертују страницу, мада није прецизна инверзија. Бројчане вредности би требало да произведу одговарајући степен ротације, самостално или у комбинацији са „rotated”, али нисам успео то да постигнем. Нити сам могао успешно да откријем чему служе „mirrored” и „negative”.

Други аргумент команде `\setuppapersize`, за који сам већ напоменуо да се може изоставити када је величина штампе иста као величина словослагања, може да узме потпуно исте вредности као и први, и означава величину папира и оријентацију. Његова вредност такође може бити и „oversized” – што према ConTeXt викију – додаје центиметар и по сваком углу папира.

Према викију, постоје и остале могуће вредности за други аргумент: „undersized”, „doublesized” и „doubleoversized”. Али у мојим личним тестовима нисам видео никакву промену када се употреби било која од ових вредности; а ни званична дефиниција команде (погледајте [одељак 3.6](#)) не помиње ове опције.

5.1.2 Употреба нестандартних величина странице

Ако желимо да употребимо нестандартну величину странице, морамо да урадимо две ствари:

1. Употребимо нестандартну синтаксу команде `\setuppapersize` која нам омогућава да наведемо висину и ширину папира као димензије.
2. Дефинишемо нашу сопствену величину странице, да јој доделимо име и да је користимо као да је у питању стандардна величина папира.

Алтернативна синтакса команде `\setuppapersize`

Уз синтаксу коју смо већ видели, `\setuppapersize` нам омогућава да употребимо и следећу:

`\setuppapersize[Име][Опције]`

где је *Име* необавезни аргумент који представља име које се командом `\definepapersize` (коју ћемо представити следећу) додељује величини папира, а *Опције* су врста аргумента којим експлицитно додељујемо вредност. Међу дозвољеним вредностима можемо да истакнемо следеће:

- **width, height** које представљају редом ширину и висину странице.
- **page, paper**. Прва се односи на величину странице на коју се слаже текст, а друга на величину странице на коју ће се физички штампати. То значи да је „page” исто што и први аргумент команде `\setuppapersize` у њеној уобичајеној синтакси (која је објашњена изнад) и „paper” као други аргумент. Ове опције могу имати исте вредности као што је раније речено (A4, S3, итд.).
- **scale**, наводи фактор скалирања за страницу.
- **topspace, backspace, offset**, додатни размаци.

Дефинисање прилагођене величине странице

За дефинисање прилагођене величине странице, користимо команду `\definepapersize` чија је синтакса

```
\definepapersize[Име][Опције]
```

где се *Име* односи на име које се даје новој величини, а *Опције* могу бити:

- Било која од дозвољених вредности за `\setuppapersize` у њеној уобичајеној синтакси (A4, A3, B5, CD, итд.).
- Мере за ширину (папира), висину (папира) и померај, или вредност скалирања („scale”).

Није могуће да се мешају вредности дозвољене за `\setuppapersize` са мерама или факторима скалирања. Разлог за то је што су у првом случају опције симболичке речи, а у другом променљиве којима се дају конкретне вредности; а у систему ConTeXt, као што сам већ рекао, није дозвољено мешање ове две врсте опција.

Када користимо `\definepapersize` да назначимо величину папира која се подудара са неким од стандардних величина, уместо да дефинишемо нову величину папира, ми ћемо уствари да дефинишемо ново име за већ постојећу величину папира. Ово може да буде корисни ако желимо да комбинујемо величину папира са оријентацијом. Тако на пример, можемо да напишемо

```
\definepapersize[vertical][A4, portrait]  
\definepapersize[horizontal][A4, landscape]
```

5.1.3 Промена величине папира на било ком месту у документу

У већини случајева документи користе само једну величину папира и то је разлог што је команда као што је `\setuppapersize` типична команда која се поставља у преамбулу и која се користи само једном у сваком документу. Међутим, у неким ситуацијама може бити неопходно да се на неком месту у документу промени величина; на пример, ако од се неког места надаље умеће анекс чији су листови положени.

У таквим случајевима можемо употребити `\setuppapersize` тачно на месту где та промена треба да се догоди. Али пошто би се промена тренутно применила, обично се пре команде `\setuppapersize` умеће форсирани прелом странице, па се тако спречавају нежељени резултати.

Ако је потребно да се величина промени само је појединачну страницу, уместо да команду `\setuppapersize` употребимо двапут, једном да подесимо нову величину, а други пут да је вратимо на оригиналну, можемо да употребимо команду `\adaptpapersize` која мења величину странице, а касније је аутоматски ресетује на вредност која је важила пре позива команде. Исто као и са `\setuppapersize`, пре употребе `\adaptpapersize` би требало да уметнемо форсирани прелом странице.

5.1.4 Подешавање величине странице према садржају

Постоје три окружења у систему ConTeXt која генеришу странице довољне величине тако да на њих стане предвиђени садржај. То су `\startMPpage`, `\startpagefigure` и `\startTEXpage`. Прво генерише страницу која садржи графику дефинисану у MetaPost, језиком за графички дизајн који се интегрише са ConTeXt, али којим се нећу бавити у овом уводу. Друго ради исто са сликом и можда са нешто текста испод ње. Оно узима сва аргумента: први идентификује фајл који садржи слику. О овоме ћу говорити у поглављу посвећеном спољним сликама. Треће (`\startTEXpage`) садржи текст који је њен садржај на страници. Њена синтакса је:

```
\startTEXpage[Опције] ... \stopTEXpage
```


где Опције могу бити било шта од следећег:



- **strut**. Нисам сигуран у вези корисности ове опције. У ConTeXt терминологији, *strut* је карактер који нема ширину, али има максималну висину и дубину, мада ми није јасно какве то везе има свеукупном употребном вредности ове команде. Према викију, ова опција дозвољава вредности „yes”, „no”, „global” и „local”, при чему је подразумевана вредност „no”.
- **align**. Назначава поравнање текста. Ово може бити „normal”, „flushleft”, „flushright”, „middle”, „high”, „low” или „lohi”.
- **offset** за назначаване количине празног простора око текста. Може бити „none”, „overlay” у случају да желите ефекат прекривања, или тачну димензију.
- **width, height** за које можемо навести ширину и висину странице, или вредност „fit” тако да ширина и висина постану такве да одговарају тексту који се налази у окружењу.
- **frame** која је подразумевано „off”, али може да узме вредност „on” ако желимо оквир око текста на страници.

5.2 Елементи на страници

ConTeXt препознаје различите елементе на страницама и њихове димензије се конфигуришу командом `\setuplayout`. Убрзо ћемо се позабавити овим, али најбоље би било да пре тога опишемо сваки од елемената странице, и да наведемо име под којим их команда `\setuplayout` познаје:

- **Ивице**: празан простор око површине текста. Мада их већина текст процесора назива „маргине”, пожељно је да се користи ConTeXt терминологија, јер нам омогућава да направимо разлику између ивица у ужем смислу, тамо где нема текста (мада се у електронским документима ту могу налазити дугмад за навигацију и слично), и маргине у којима се понекада могу налазити текст елементи, као што су на пример, белешке на маргинама.
 - Висина горње ивице се контролише помоћу две мере: самом горњом ивицом („top”) и растојањем између горње ивице и заглавља („topdistance”). Збир ове две мере се назива „topspace”.
 - Величина леве и десне ивице зависи од мера „leftedge” и „rightedge” редом. Ако желимо да и једна и друга имају исту дужину, можемо истовремено да их подесимо опцијом „edge”.

У документима намењеним за двострану штампу, не говоримо о левој и десној ивици, већ о унутрашњој и спољашњој; прва је ивица најближа месту на којем ће се листови хефати или прошивати, тј. лева ивица на непарно нумерисаним страницама, а десна ивица парним страницама. Спољашња ивица је насупрот унутрашње.

 - Висина ниже ивице се назива „bottom”.
- **Маргине** у ужем смислу. ConTeXt маргинама назива само бочне маргине (леву и десну). Маргине се налазе између ивице и простора главног текста и предвиђено је да се у њима налазе одређени текст елементи као што су, на пример, белешке на маргинама, наслови одељака или њихови бројеви.

Димензије које контролишу величину маргина су:

- **margin**: користи се када истовремено желимо да поставимо маргине на исту величину.
- **leftmargin, rightmargin**: постављају величину леве и десне маргине, редом.
- **edgedistance**: растојање између ивице и маргине.

- **leftedgedistance, rightedgedistance:** растојање између ивице и леве и десне маргине, редом.
- **margindistance:** растојање између маргине и области главног текста.
- **leftmargindistance, rightmargindistance:** растојање између области главног текста и леве и десне маргине, редом.
- **backspace:** ова мера представља размак између левог угла папира и почетка области главног текста. Дакле, она представља збир „leftedge” + „leftedgedistance” + „leftmargin” + „leftmargindistance”.
- **Заглавље и подножје:** заглавље и подножје странице су две области које се редом налазе на изнад и испод површине странице по којој се пише. Обично садрже информације које помажу да се текст одреди контекст текста, као што су бројеви страница, име аутора, наслов дела, наслов поглавља или одељка, итд. На ове делове странице се у систему ConTeXt утиче следећим димензијама:
 - **header:** висина заглавља.
 - **footer:** висина подножја
 - **headerdistance:** растојање од заглавља до области главног текста на страници.
 - **footerdistance:** растојање између подножја и области главног текста на страници.
 - **topdistance, bottomdistance:** већ су раније поменуте. Представљају растојање између горње ивице и заглавља или доње ивице и подножја, редом.
- **Главна област текста:** ово је најшира област странице у којој се налази текст документа. Његова величина зависи од променљивих „width” и „textheight”. Променљива „height”, представља збир „header”, „headerdistance”, „textheight”, „footerdistance” и „footer”.

Све ове области можемо видети на [слици 5.1](#) заједно са именима одговарајућих мера и стрелицама које означавају докле се простиру. Дебљина стрелице заједно са величином имена стрелица одговара важности сваке од ових раздаљина за распоред странице. Ако пажљиво погледамо, видећемо да ова слика приказује да се страница може представити као табела са 9 редова и 9 колона, или ако не узмемо у обзир вредности размака између различитих области, онда је то пет редова и пет колона од којих текст може да буде само у три реда и три колоне. Пресек средњег реда са средњом колоном чини област главног текста и обично ће чинити већи део странице.

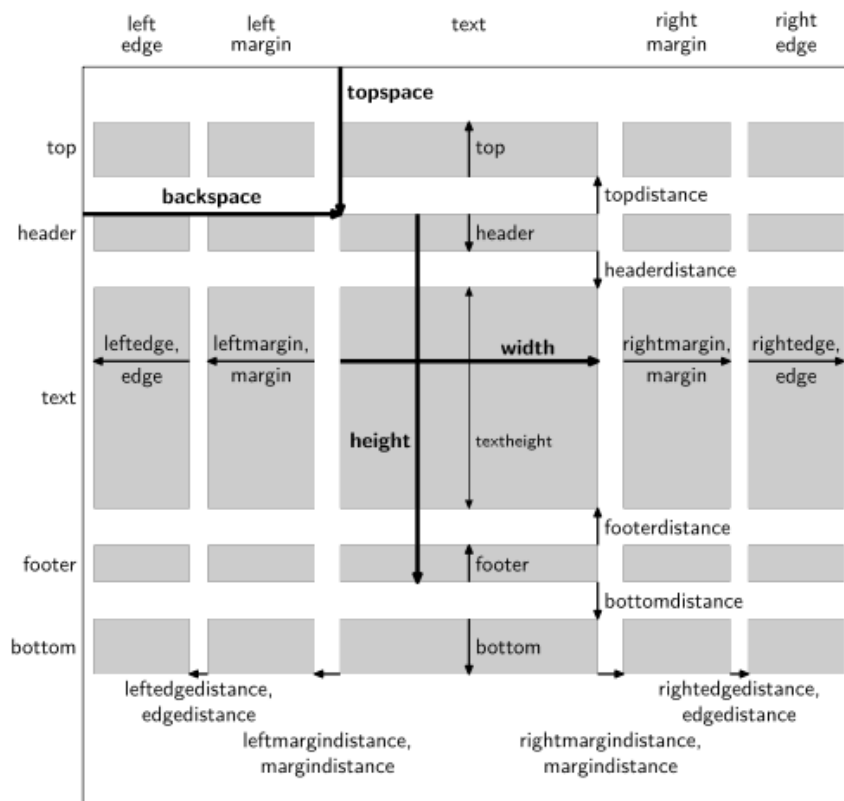
У фази креирања распореда нашег документа, све мере странице можемо видети командом `\showsetups`. Ако желите да на страници видите визуелну представу обриса дистрибуције текста, можете да употребите команду `\showframe`; а командом `\showlayout` можете видети комбинацију претходне две команде.

5.3 Распоред странице (`\setuplayout`)

5.3.1 Додела величине различитим компонентама странице

Дизајн странице подразумева доделу одређених величина одговарајућим областима странице. Ово се ради командом `\setuplayout`. Она нам омогућава да изменимо било коју од димензија поменутих у претходном одељку. Њена синтакса је следећа:

```
\setuplayout[Name][Options]
```



Слика 5.1 Области и мере на страници

где је *Име* необавезни аргумент који се користи само у случају када дизајнирамо више распореда (погледајте одељак 5.3.3), а опције су, поред осталих које ћемо видети касније, било које од претходно поменутих. Међутим, имајте на уму да су ове мере у међусобној вези, јер укупан збир компоненти које утичу на ширину, оних које утичу на висину мора да се подудари са ширином и висином странице. Ово у суштини значи да када мењамо неку хоризонталну дужину, морамо да променимо и остатак хоризонталних дужина; а исто је и када подешавамо вертикалну дужину.

ConTeXt подразумевано извршава аутоматска подешавања димензија само у неким случајевима, који нису на потпун или систематичан начин наведени у његовој документацији. Извођењем неколико тестова сам потврдио, на пример, да ручно увећање или смањење висине заглавља или подножја повлачи и измену „textheight”; међутим, ручна измена неке од маргина не подешава аутоматски и ширину текста, „width” (барем је то резултат мојих тестова). Због тога је најефикаснији начин за спречавање неконзистентности између величине странице (постављене са `\setuppapersize`) и величине њених одговарајућих компоненти следећи:

- Што се тиче хоризонталних мера:
 - Подешавањем „backspace” (које укључује „leftedge” и „leftmargin”).
 - Подешавањем „width” (ширине текста) не са димензијама, већ са вредностима „fit” или „middle”:
 - * fit израчунава ширину текста на основу ширине осталих хоризонталних компоненти странице.
 - * middle ради исто, али најпре поставља десну и леву маргину на исту вредност.

- Што се тиче вертикалних мера:
 - Подешавањем „`topspace`”.
 - Подешавањем „`fit`” or „`middle`” values to „`height`”. Оне раде на исти начин као код ширине. Прва израчунава висину према осталим компонентама, а друга прво поставља горњу и доњу маргину на исту вредност, па онда израчунава висину текста.
 - Када се подеси „`height`”, по потреби се подешавају висина заглавља или подножја, знајући да ће се „`textheight`” у тим случајевима аутоматски поново подесити.
- Још једна могућност за индиректно одређивање висине области главног текста је навођењем броја линија које треба да стану у њу (имајући у виду текући размак између линија и величину фонта). Ово је разлог што `\setuplayout` поседује „`lines`” опцију.

Постављање логичке странице на физичку страницу

У случајевима када величина логичке странице није иста као величина физичке странице (погледајте одељак 5.1.1) `\setuplayout` нам омогућава да конфигуришемо неке додатне опције које утичу на постављање логичке странице на физичку:

- **location:** ова опција одређује место на физичкој страници на којем ће се поставити логичка. Могуће вредности су `left`, `middle`, `right`, `top`, `bottom`, `singlesided`, `doublesided` или `duplex`.
- **scale:** наводи фактор скалирања странице пре постављања на физичку страницу.
- **marking:** на страници ће се штампати визуелни маркери који показују где се сече папир.
- **horoffset, veroffset, clipoffset, cropoffset, trimoffset, bleedoffset, artoffset:** низ мера које наводе различите помераје на физичкој страници. Већина њих је објашњена у референтном упутству из 2013. године.

Ове `\setuplayout` опције морају да се комбинују са назнакама из `\setuparranging` које говоре како се на физичкој страници ређају логичке странице. Ове команде нећу да описујем у уводу јер их уопште нисам тестирао.

Добијање ширине и висине области текста

Команде `\textwidth` и `\textheight` редом враћају ширину и висину области текста. Вредности које се добију помоћу њих не могу директно да се покажу у финалном документу, али могу да се употребе за постављање мера висине и ширине осталих команди. На пример, ако желимо да наведемо слику чија ће ширина бити 60% ширине линије, као вредност опције „`width`” за слику морамо да поставимо: „`width=0.6\textwidth`”.

5.3.2 Подешавање распореда странице

Може се догодити да наш распоред странице на одређеној страници не даје жељени резултат; на пример, последња страница поглавља са само једом или две линије, што није пожељно ни типографски, ни естетски. За решавање оваквих ситуација, ConTeXt обезбеђује команду `\adaptlayout` која нам омогућава да променимо величину области текста на једној или на више страница. Предвиђено је да се ова команда користи само онда када смо завршили са писањем документа и радимо само ситна финална подешавања. Стога, њена природна позиција је у преамбули документа. Синтакса команде је:

```
\adaptlayout[Странице] [Опције]
```

где се *Странице* односи на број странице или страница чији распоред желимо да променимо. То није обавезни аргумент и он би требало да се користи само када се команда `\adaptlayout` поставља

у преамбулу. Можемо да назначимо само једну страницу, или неколико њих, тако што бројеве раздвајамо запетама. Ако изоставимо овај први аргумент, `\adaplayout` ће утицати само на страницу која садржи команду.

Опције могу да буду:

- **height**: дозвољава нам да наведемо, као меру, висину странице у питању. Можемо да задамо апсолутну висину (нпр. „19cm”) или релативну висину (нпр. „+1cm”, „-0.7cm”).
- **lines**: можемо да наведемо број линија који треба да се дода или одузме. Линије се додају ако се испред броја стави +, а одузимају ако се постави знак – (не просто цртица).

Имајте на уму да када мењамо број линија на страници, такође може да се промени и пагинација остатка документа. Из тог разлога је пожељно да се команда `\adaplayout` користи само на крају, када више нема накнадних измена документа, и да се то уради у преамбули. Затим одлазимо на прву страницу коју желимо да изменимо, извршимо измену, па проверимо како то утиче на наредне странице; ако је измена таква да још нека страница мора да се промени, додајемо и њен број па компајлирамо још једном, и тако даље.

5.3.3 Употреба више различитих распореда страница

Ако је потребно да за различите делове документа употребимо различите распореде странице, најбољи начин да се почне је да се дефинише *општи* распоред, па затим разни алтернативни који мењају само димензије које се разликују од оних у општем. Ови алтернативни распореди наслеђују све особине општег распореда које се не мењају као део своје дефиниције. За навођење алтернативног распореда, његово називање именом којим га касније можемо позвати, користимо команду `\definelayout` чија је општа синтакса:

```
\definelayout[Име/Број] [Конфигурација]
```

где је *Име/Број* име које се придружује новом дизајну, или број странице на којем се нови распоред аутоматски активира, а *Конфигурација* садржи аспекте распореда које желимо да променимо у односу на општи распоред.

Када се новом распореду придружи име, на одређеном месту у документу га позивамо помоћу:

```
\setuplayout[ИмеРаспореда]
```

а враћамо се на општи распоред са:

```
\setuplayout[reset]
```

С друге стране, ако је нови распоред придружен одређеној страници, он ће се аутоматски активирати када се достигне та страница. Међутим, једном када се активира, враћање на општи дизајн се ради експлицитно, мада ово можемо да *полуаутоматизујемо*. На пример, ако желимо да применимо распоред само на странице 1 и 2, у преамбули документа можемо да напишемо следеће:

```
\definelayout[1][...]  
\definelayout[3][reset]
```

Резултат ових команди је да се распоред дефинисан у првој линији активира на страници 1, а на страници 3 се активира други распоред чија је функција просто враћање на општи распоред.

Са `\definelayout[even]` креирамо распоред који се активира на свим парним страницама; а са `\definelayout[odd]` распоред који се примењује на све непарне странице.

5.3.4 Остале ствари у вези распореда странице

А. Разликовање непарних и парних страница

У документима припремљеним за двострану штампу је чест случај да се заглавље, нумерација страница и бочне маргине разликују на непарним и парним страницама. Парно нумерисане странице се

такође називају и леве (версо) странице, а непарне странице десне (ректо) странице. У овим случајевима је такође обичај да се мења терминологија која се тиче маргина, па говоримо о унутрашњим и спољашњим маргинама. Ове раније се налазе на тачки најближој месту на којем ће се странице хефати или прошивати, а касније на супротној страни. На непарно нумерисаним страницама, унутрашња маргина одговара левој маргини, а на парно нумерисаним спољашња маргина одговара десној маргини.

`\setuplayout` нема ниједну опцију која нам омогућава да директно кажемо да желимо разлику између распореда парних и непарних страница. То је зато што се систему ConTeXt разлика између две врсте страница поставља другачијом опцијом: `\setuppagenumbering` коју ћемо видети у одељку 5.4. Када се ово подеси, ConTeXt претпоставља да је страница описана са `\setuplayout` непарна страница и парну страницу изграђује тако што на њу примени инвертоване вредности за непарну страницу: спецификације које се примењују за непарно нумерисане странице се примењују на леву, а на парно нумерисаним страницама на десну; и обрнуто: оне које се примењују на непарно нумерисану страницу на десну, у парно нумерисаној страници се примењују на леву.

Б. Странице са више од једне колоне

Помоћу `\setuplayout` текст нашег документа такође можемо да видимо распоређен у две или више колона, као у неким новинама и магазинима, на пример. Ово се контролише опцијом „columns” чија вредност мора да буде цео број. Када постоји више од једне колоне, размак између њих се задаје опцијом „columndistance”.

Ова опција је намењена за документе у којима је сав текст (или већи део текста) распоређен у више колона. Ако у документу који је углавном у једној колони пожељимо да одређени део буде у две или три колоне, не морамо да мењамо распоред странице, већ просто употребимо „columns” окружење (погледајте одељак 12.2).

5.4 Нумерација страница

ConTeXt подразумевано користи арапске бројеве за нумерацију страница и број се појављује у заглављу, хоризонтално центриран. Ако ове особине желите да промените, ConTeXt вам нуди различите процедуре за то, које према мом мишљењу, ствар чине непотребно компликованом.

Најпре, најосновније карактеристике нумерације се контролишу помоћу две различите команде: `\setuppagenumbering` и `\setupuserpagenumber`.

`\setuppagenumbering` нуди следеће опције:

- **alternative:** ова опција контролише да ли је документ дизајниран тако да су заглавље и подножје идентични на свим страницама („singlesided”), или се прави разлика између парних и непарних страница („doublesided”). Када ова опција има последњу вредност, аутоматски се утиче на вредности уведене са „setuplayout”, тако да се претпоставља да се подешавање „setuplayout” односи само на непарно нумерисане странице, па дакле, оно што се постави за леву маргину уствари је за унутрашњу (што је на парно нумерисаним страницама на десној страни) и да се оно што се постави за десну страну уствари односи на спољашњу маргину, која је на парно нумерисаним страницама на левој страни.
- **state:** означава да ли ће се број странице приказивати, или не. Могуће су две вредности: start (број странице се приказује) и stop (бројеви страница се не приказују). Имена ових вредности (старт и стоп) би могла да нас наведу на помисао да када имамо „state=stop” странице се не нумеришу, а када је „state=start” нумерација поново почиње. Али није тако: ове вредности утичу само на то да ли се број приказује или не.
- **location:** означава где ће се приказивати број. Обичне је потребно да наведемо две вредности у овој опцији, раздвојене запетом. Најпре морамо да наведемо да ли желимо број странице у заглављу („header”), или у подножју („footer”), па затим, где: то може да буде „left”, „middle”,

„right”, „inleft”, „inright”, „margin”, „inmargin”, „atmargin” или „marginedge”. На пример: ако желимо да се прикаже десно поравната нумерација у подножју, требало би да наведемо „location={footer,right}”. Иначе, погледајте како смо ову опцију поставили унутар витичастих заграда, тако да ConTeXt исправно може да интерпретира раздвајајућу запету.

- **style**: назначавља величину фонта и стил који треба да се користи за бројеве страница.
- **color**: назначавља боју која треба да се примени на број странице.
- **left**: прихвата команду или текст који треба да се изврши лево од броја странице.
- **right**: прихвата команду или текст који треба да се изврши десно од броја странице.
- **command**: прихвата команду којој ће се број странице проследити као параметар.
- **width**: назначавља ширину коју заузима број странице.
- **strut**: нисам сигуран у вези овога. У мојим тестовима, када је „strut=no”, број се штампа тачно на горњој ивици заглавља или на дну подножја, док када је „strut=yes” (подразумевана вредност) између броја и странице се примењује размак.

`\setupuserpagenumber` прихвата и следеће додатне опције:

- **numberconversion**: контролише врсту набрајања која може бити арапска („n”, „numbers”), мала слова („a”, „characters”), велика слова („A”, „Characters”), капитал („KA”), римско малим („i”, „r”, „romannumerals”), римско великим („I”, „R”, „Romannumerals”) или римско капиталом („KR”).
- **number**: наводи број који се додељује првој страници, на основу којег се израчунавају остали.
- **numberorder**: ако овоме као вредност доделимо „reverse”, нумерација страница ће се вршити у опадајућем редоследу; то значи да ће последња страница бити 1, претпоследња 2, итд.
- **way**: омогућава нам да одредимо како ће се наставити са нумерацијом. Може бити: byblock, bychapter, bysection, bysubsection, итд.
- **prefix**: дозвољава нам да наведемо префикс бројевима страница.
- **numberconversionset**: објашњено је испод.

Уз ове две команде, такође је неопходно да се узме обзир контрола бројева која се тиче макроструктуре документа (погледајте одељак 7.6). Са ове тачке гледишта, `\defineconversionset` нам омогућава да наведемо другачију врсту нумерације са сваки макроструктурни блок. На пример:

```
\defineconversionset
  [frontpart:pagenumber][romannumerals]

\defineconversionset
  [bodypart:pagenumber][numbers]

\defineconversionset
  [appendixpart:pagenumber][Characters]
```

ће уредити да се први блок нашег документа (frontmatter) броји римским бројевима исписаним малим словима, средишњи блок (bodymatter) арапским бројевима, а додаци великим словима.

Следећим командама можемо добити број странице:

- `\userpagenumber`: враћа број странице онакав како је конфигурисан командама `\setuppagenumbering` и `\setupuserpagenumber`.
- `\pagenumber`: враћа исти број као претходна команда, али још увек арапским бројевима.
- `\realpagenumber`: враћа реални број странице у арапским бројевима без потребе било које од ових спецификација.

Ако желимо да добијемо број последње странце у документу, постоји три команде које одговарају претходним. То су: `\lastuserpagenumber`, `\lastpagenumber` и `\lastrealpagenumber`.

5.5 Форсирани или предложени преломи странице

5.5.1 Команда `\page`

Алгоритам за распоред текста у систему ConTeXt је прилично сложен и заснива се на многим израчунавањима и интерним променљивама које програму говоре где се из перспективе типографске коректности налази најбоља тачка за уметање стварног прелома странице. Команда `\page` нам омогућава да утичемо на овај алгоритам:

а. Сугерисањем одређених тачака као најбољих или најгорих места за уметање прелома странице.

- **no**: наводи да место на којем се налази команда није добар кандидат за уметање прелома странице, тако да би прелом требало да се уметне на неком другом месту у документу, што је даље могуће.
- **bigpreference**: наводи да је тачка на којој се наиђе на команду *врло добро место* да се покуша прелом странице, али не толико да се прелом форсира.

Запазите да ове три опције нити форсирају, нити спречавају преломе страница, већ само говоре систему ConTeXt да би требало да узме у обзир оно што је наведено у команду када тражи најбоље место за прелом. Међутим, у крајњој линији, ConTeXt ће и даље одлучивати о месту на којем ће се уметнуто прелом странице.

б. Форсирањем прелома странице на одређеном месту; у овом случају можемо такође да назначимо колико прелома страница би требало да се направи, као и одређене особине страница које се умећу.

- **yes**: форсира прелом странице на овом месту.
- **makeup**: слично као и „yes“, али форсирани прелом је ради одмах, не поставља се пре тога ниједан плутајући објекат који чека на постављање (погледајте одељак 13.1).
- **empty**: умеће потпуно празну страницу у документ.
- **even**: умеће потребан број страница тако да наредна страница постане парна страница.
- **odd**: умеће потребан број страница тако да наредна страница постан непарна страница.
- **left**, **right**: слично као и претходне две опције, али се примењује само на документе предвиђене за двострану штампу, са различитим заглављима, подножјима или маргинама у зависности од тога да ли је страница парна или непарна.
- **quadruple**: умеће потребан број страница потребан да наредна страница буде умножак броја 4.

Уз ове опције које су намењене директној контроли пагинације, `\page` нуди и остале опције које утичу на начин функционисања ове команде. Истичу се опција „disable“ која чини да ConTeXt игнорише `\page` команде на које наиђе од овог места па надаље, као и опција „reset“ која има супротан ефекат, која дакле враћа ефекат наредних `\page` команди на које се наиђе.

5.5.2 Спајање одређених линија пасуса тако да се спречи уметање прелома странице између њих

Понекада када желимо да спречимо прелом странице између неколико пасуса, употреба команде `\page` може бити заморна, јер би требало да се напише на свакој тачки где је могуће да се уметне прелом странице. Једноставнија процедура за то је да се материјал које желимо да буде на истој страници постави у оно што TeX назива *вертикална кутуја*.

На почетку овог документа (на [страници 16](#)) сам напоменуо да је интерно за \TeX све кутија. Појам кутије је у \TeX је основа за било коју врсту *напредне* операције; али управљање њима је сувише сложено да би се укључило у овај увод. То је разлог што кутије само повремено помињем.

Једном када се креирају, \TeX кутије су недељиве, што значи да не можемо уметнути прелом странице који би кутију поделио на две. Зато ако материјал који желимо да држимо заједно поставимо у невидљиву кутију, спречићемо уметање прелома странице који би поделио тај материјал. То се ради командом `\vbox`, чија је синтакса

```
\vbox{Материјал}
```

где је *Материјал* текст који желимо да буде заједно.

Нека \ConTeXt окружења постављају свој садржај у кутију. На пример, „`framedtext`“, тако да ако уквиримо материјал који желимо да држимо заједно у ово окружење и поставимо да је оквир невидљив (што може да се уради опцијом `frame=off`), постићи ћемо исту ствар.

5.6 Заглавља и подножја

5.6.1 Команде за постављање саржаја заглавља и подножја

Ако смо у распореду странице заглављу и подножју доделили одређену величину, у њих можемо уметнути текст командама `\setupheadertexts` и `\setupfootertexts`. Оне су сличне, једина разлика је што прва активира садржај заглавља, а друга садржај подножја. Обе имају од једног до пет аргумената.

1. Ако се употреби са једним аргументом, он ће садржати текст који ће се у заглављу или подножју поставити на средину ширине странице. На пример: `\setupfootertexts[pagenumbers]` ће на средини подножја писати број странице.
2. Ако се употреби са два аргумента, садржај првог аргумента ће се поставити на леву страну заглавља или подножја, а онај другог аргумента на десну страну. На пример, `\setupheadertexts[Предговор][pagenumbers]` ће сложити заглавље странице у којем се реч „предговор“ испишује на левој страни, а број странице на десној страни.
3. Ако се наведу три аргумента, први ће назначити *област* у којој ће се штампати преостала два. *Област* се односи на *области* странице поменуте у [одељку 5.2](#), другим речима: `edge`, `margin`, `header`... Остала два аргумента садрже текст који треба да се постави у леву ивицу, маргину, десну ивицу, маргину.

Када се користи са четири или пет аргумената, то је исто као и када се користи са два или три аргумента, само у случајевима када се прави разлика између парних и непарних страница, а који се јављају, као што знамо, када се са `\setuppagenumbering` постави „`alternative=doublesided`“. У том случају, два могућа аргумента се додају тако да означавају садржај леве и десне стране парних страница.

Важна карактеристика ове две команде је да када се користе са два аргумента, претходно централно заглавље или подножје (ако је постојало) се не преписује, што нам омогућава да у свакој области напишемо другачији текст све док прво напишемо централни текст (позивом команде са једним аргументом), па затим напишемо текстове за обе стране (тако што је позовемо поново, али сада са два аргумента). Дакле, ако напишемо следеће команде

```
\setupheadertexts[и]
\setupheadertexts[Станлио][Олио]
```

прва команда ће написати „и“ на средини заглавља, а друга ће написати „Станлио“ на левој и „Олио“ на десној страни, остављајући средишњу област нетакнутом, јер јој није наложено да је поново испише. Коначно заглавље ће се сада приказивати овако

Станлио

и

Олио



Објашњење рада ових команди које сам управо изложио је мој закључак након многих тестова. ConTeXt excursion даје објашњење ових команди засновано на верзији са пет аргумената; а оно у референтном упутству из 2013. године је засновано на верзији са три аргумената. Чини ми се да је моје јасније. С друге стране, нисам наишао на објашњење зашто други позив команде не препишује претходни позив, али то је начин на који она ради ако прво напишемо централну ставку у заглављу или подножју, па затим оне са обе стране. Али ако прво упишемо бочне ставке заглавља/подножја, наредни позив команде који уписује централну ставку ће обрисати претходна заглавља или подножја. Зашто? Немам појма. Чини ми се да ови мали детаљи уносе непотребну компликацију и требало би да се јасно објасне у званичној документацији.

Штавише, као садржај заглавља или подножја можемо да наведемо било коју комбинацију текста и команди. А такође и следеће вредности:

- **date, currentdate:** ће написати (било које од њих) текући датум.
- **pagenumber:** ће написати број странице.
- **part, chapter, section...:** ће написати наслов који одговара делу, поглављу, одељку... или било које структурне поделе која постоји.
- **partnumber, chapternumber, sectionnumber...:** ће написати број дела, поглавља, одељка... или било које структурне поделе која постоји.

Пажња: ова симболичка имена (date, currentdate, pagenumber, chapter, chapternumber, итд.) се интерпретирају као таква само ако је једини садржај аргумента то симболичко име; али ако додамо још неки текст или команду форматирања, ове речи ће се интерпретирати дословно, па тако, ако напишемо, на пример, `\setupheadertexts[chapternumber]` добићемо број текућег поглавља; али ако напишемо `\setupheadertexts[Поглавље chapternumber]` завршићемо са: „Поглавље chapternumber“. У овим случајевима, када садржај команде није само симболичка реч, морамо да:

- За date, currentdate и pagenumber употребимо не симболичку реч, већ команду са истим именом (`\date`, `\currentdate` или `\pagenumber`).
- За part, partnumber, chapter, chapternumber, итд. употребимо `\getmarking[Mark]` команду која враћа садржај Mark који се тражи. Тако ће, на пример, `\getmarking[chapter]` да врати наслов текућег поглавља, док ће `\getmarking[chapternumber]` вратити број текућег поглавља.

Ако на одређеној страни желимо да искључимо заглавља и подножја, користимо команду `\noheaderandfooterlines` која ради само на страници на којој се нађе. Ако на некој страници само желимо да обришемо број странице, морамо да употребимо команду `\page[blank]`.

5.6.2 Форматирање заглавља и подножја

Одређени формат у којем се приказује текст заглавља или подножја може да се наведе у аргументима за `\setupheadertexts` или `\setupfootertexts` користећи одговарајуће команде форматирања. Међутим, ово такође можемо и глобално да конфигуришемо са `\setupheader` и `\setupfooter` које нуде следеће опције:

- **state:** прихвата следеће вредности: start, stop, empty, high, none, normal или nomarking.
- **style, leftstyle, rightstyle:** конфигурација стила за текст заглавља или подножја. style утиче на све странице, leftstyle на парне странице, а rightstyle на непарне странице.
- **color, leftcolor, rightcolor:** боја заглавља или подножја. Може да утиче на све странице (color опција) или само на парне странице (leftcolor) или непарне странице (rightcolor)
- **width, leftwidth, rightwidth:** ширина свих заглавља и подножја (width) или заглавља/подножја на парним страницама (leftwidth) или непарним (rightwidth).

- **before:** команда која треба да се изврши пре писања заглавља или подножја.
- **after:** команда која ће се извршити након писања заглавља или подножја.
- **strut:** ако има вредност „yes”, појављује се вертикални раздвајајући простор између заглавља и ивице. Када има вредност „no”, заглавље или подножје додирује области горње и доње ивице.

5.6.3 Дефинисање специфичних заглавља и подножја и везивање за section команде

ConTeXt систем заглавља и подножја нам омогућава да се текст у заглављу или подножју аутоматски промени када мењамо поглавља или одељке; или када мењамо странице, у случају да имамо различити сет заглавља или подножја за парне и непарне странице. Али он нам не омогућава да правимо разлику између прве странице (документа, или поглавља или одељка) и осталих страница. Да бисмо то постигли, морамо:

1. Да дефинишемо специфично заглавље или подножје
2. Да га повежемо са одељком на који се односи.

Дефинисање специфичних заглавља или подножја се врши командом `\definertext`, чија је синтакса:

```
\definertext
  [Име] [Тип]
  [Садржај1] [Садржај2] [Садржај3]
  [Садржај4] [Садржај5]
```

где је *Име* име које се додељује заглављу или подножју са којим радимо; *Тип* може бити `header` или `footer`, у зависности од тога шта дефинишемо, а осталих пет аргумената држе садржај новог заглавља или подножја, на сличан начин као што смо видели код `\setupheadertexts` и `\setupfootertexts`. Кад то урадимо, потребно је да ново заглавље или подножје повежемо са неким одређеним одељком помоћу `\setuphead` користећи опције `header` и `footer` (које су описане у [поглављу 7](#)).

Дакле, наредни пример ће да сакрије заглавље на првој страници сваког поглавља, а у подножју ће се појавити центриран број странице:

```
\definertext[ChapterFirstPage] [footer] [pagenumber]
\setuphead
  [chapter]
  [header=high, footer=ChapterFirstPage]
```

5.7 Уметање текст елемената у ивице и маргине странице

Горња и доња ивица, као и десна и лева маргина обично не садрже никакав текст. Међутим, ConTeXt омогућава да ту поставимо неке текст елементе. Тачније, за то су доступне следеће команде:

- `\setuptoptexts`: омогућава нам да поставимо текст у горњу ивицу странице (изнад области заглавља).
- `\setupbottomtexts`: омогућава нам да поставимо текст у доњу ивицу странице (испод области подножја).
- `\margintext`, `\atleftmargin`, `\atrightmargin`, `\ininner`, `\ininneredge`, `\ininnermargin`, `\inleft`, `\inleftedge`, `\inleftmargin`, `\inmargin`, `\inother`, `\inouter`, `\inouteredge`, `\inoutermargin`, `\inright`, `\inrightedge`, `\inrightmargin`: дозвољавају постављање текста у бочне ивице и маргине документа.

Прве две команде раде исто као `\setupheadertexts` и `\setupfootertexts` и формат ових текстова може чак унапред да се конфигурише помоћу `\setuptop` и `\setupbottom`, слично као што нам `\setupheader` дозвољава да конфигуришемо текстове за `\setupheadertexts`. У вези свега овога, укажујем на оно што сам већ рекао у одељку 5.6. Само је потребно је да се дода мали детаљ да текст који се подеси са `\setuptoptexts` или `\setupbottomtexts` неће бити видљив ако се у распореду странице не резервише простор за горњу (top) или доњу (bottom) ивицу. У вези тога, погледајте одељак 5.3.1.

Што се тиче команди намењених постављају текста у маргине документа, све оне имају сличну синтаксу:

`\ИмеКоманде[Референца][Конфигурација]{Текст}`

где *Референца* и *Конфигурација* нису обавезни аргументи; први се користи са могуће унакрсно референцирање, а други нам омогућава да подесимо маргинални текст. Последњи аргумент постављен у витичасте заграде садржи текст који се поставља у маргину.

Општија команда од ових овде је `\margintext` јер омогућава постављање текста у било коју маргину или бочну ивицу странице. Остале команде, као што њихово име наговештава, постављају текст на саму маргину (десну или леву, унутрашњу или спољашњу). Оне су уско везане за распоред странице, јер ако на пример, употребимо `\inrightrightedge` а нисмо резервисали довољно простора у распореду странице за десну ивицу, неће се видети ништа.

Конфигурационе опције за `\margintext` су следеће:

- **location:** наводи у коју маргину ће се сместити текст. Може бити `left`, `right` или, у документима за двострану штампу, `outer` или `inner`. Подразумевано је `left` за документе предвиђене за једнострану штампу, а `outer` у онима за двострану.
- **width:** ширина доступна за штампање текста. Подразумевано је то пуна ширина маргине.
- **margin:** наводи да ли се текст поставља у саму маргину (`margin`) или у ивицу (`edge`).
- **align:** поравнање текста. Овде се користе исте вредности као за `\setupalign 11.6.1`.
- **line:** омогућава да наведемо број линија за који ће се померити текст у маргини. Тако да `line=1` значи да ће се текст померити једну линију наниже, а `line=-1` за једну линију навише.
- **style:** команда или команде које наводе стил текста који се поставља у маргине.
- **color:** боја маргиналног текста.
- **command:** име команде којој ће се текст намењен маргини проследити као аргумент. Ова команда ће се извршити пре исписивања текста. На пример, ако желимо да исцртамо оквир око текста, могли бисмо да употребимо „`[command=\framed]{Text}`”.

Остале команде прихватају исте опције, осим за `location` и `margin`. Тачније, команде `\atrightmargin` и `\atleftmargin` постављају текст потпуно наслоњен на тело странице. Простор раздвајања можемо да подесимо опцијом `distance`, коју нисам поменуо када сам говорио о `\margintext` јер приликом својих тестова нисам приметио никакав ефекат на ту команду.



Уз горе поменуте опције, ове команде подржавају и остале опције (`strut`, `anchor`, `method`, `category`, `scope`, `option`, `hoffset`, `voffset`, `dy`, `bottomspace`, `threshold` и `stack`) које нисам поменуо јер нису документоване, а искрено, нисам ни сигуран чему служе. За оне са именима као што је `distance` можемо да погодимо, али шта је са осталима? Вики помиње само опцију `stack`, за коју каже да се користи за емуляцију команде `\marginpars` у \LaTeX , али то ми баш и није јасно.

Команда `\setupmargindata` нам омогућава да глобално конфигуришемо текстове у свакој маргини. Тако, на пример

`\setupmargindata[right][style=slanted]`

обезбеђује да се сви текстови у десној маргини исписују косим слогом.

Такође можемо да креирамо и прилагођену команду са

```
\definemargindata[Име][Конфигурација]
```

Глава 6

Фонтови и боје у системе

ConTeXt

Садржај: 6.1 Типографски фонтови укључени у „ConTeXt Standalone“; 6.2 Особине фонтова; 6.2.1 Фонтови, *стилови* и стилске варијанте; 6.2.2 Величина фонтова; 6.3 Постављање главног фонтова документа; 6.3.1 Како изгледа фонт; 6.4 Промена неких особина фонтова; 6.4.1 Команде `\setupbodyfont` и `\switchtobodyfont`; 6.4.2 Брза измена стила, алтернативе и величине; 6.4.3 Дефинисање команди и кључних речи за величине фонтова, стилове и алтернативе; 6.5 Остале ствари везане за употребу неких алтернатива; 6.5.1 Курзив, коса слова и наглашавање; 6.5.2 Капитал и лажни капитал; 6.6 Употреба и конфигурација боја; 6.6.1 Процедуре за слагање фрагмената текста у боји; 6.6.2 Промена боје позадине и предњег плана документа; 6.6.3 Комадне за бојење одређених фрагмената текста; 6.6.4 Предефинисане боје; 6.6.5 Да видимо доступне боје; 6.6.6 Дефинисање сопствених боја;

6.1 Типографски фонтови укључени у „ConTeXt Standalone“

ConTeXt систем за фонтове пружа многе могућности, али је такође и прилично сложен. У овом упутству нећу анализирати све напредне могућности, већ ћу се ограничити на претпоставку да радимо са неким од 21 фонтова који се испоручују у ConTeXt Standalone инсталацији, онима приказаним у табели 6.1.

| Званично име | Име(на) у ConTeXt | Пример |
|-----------------------|---------------------------------|--|
| Latin Modern | modern, modern-base | Emily Brontë's book |
| Antykwa Poltawskiego | antypolawskiego | Emily Brontë's book |
| Antykwa Toruńska | antykwa | Emily Brontë's book |
| Cambria | cambria | Књига Емили Бронте Emily Brontë's book |
| DejaVu | dejavu | Књига Емили Бронте Emily Brontë's book |
| DejaVu Condensed | dejavu-condensed | Књига Емили Бронте Emily Brontë's book |
| Gentium | gentium | Књига Емили Бронте Emily Brontë's book |
| Iwona | iwona | Emily Brontë's book |
| Latin Modern Variable | modernvariable, modern-variable | Emily Brontë's book |
| PostScript | postscript | Emily Brontë's book |
| TeX Gyre Adventor | adventor, avantgarde | Emily Brontë's book |
| TeX Gyre Bonum | bonum, bookman | Emily Brontë's book |
| TeX Gyre Cursor | cursor, courier | Emily Brontë's book |
| TeX Gyre Heros | heros, helvetica | Emily Brontë's book |
| TeX Gyre Schola | schola, schoolbook | Emily Brontë's book |
| TeX Gyre Chorus | chorus, chancery | Emily Brontë's book |
| TeX Gyre Pagella | pagella, palatino | Emily Brontë's book |
| TeX Gyre Termes | termes, times | Emily Brontë's book |
| Euler | eulernova | Emily Brontë's book |
| Stix2 | stixtwo | Књига Емили Бронте Emily Brontë's book |
| Xits | xits | Књига Емили Бронте Emily Brontë's book |

Табела 6.1 Фонтови који су део ConTeXt дистрибуције

Средишња колона [табеле 6.1](#) наводи име или имена под којим ConTeXt познаје одређени фонт. Када има два имена, она су синоними. Последња колона приказује како изгледа фонт. Они који подржавају

ћирилицу имају исписан пример и ћирилицом. Што се тиче редоследа у којем су приказани, први је фонт који ConT_EXt подразумевано користи, а остали су поређани по абеди, док су последња три фонтови дизајнирани специјално за математику. Обратите пажњу да Euler фонт не може директно да прикаже акцентована слова, тако да видимо Brontë's уместо Brontë's.

За читаоце који долазе из Windows света и познају његове подразумеване фонтове, нагласићу да је *heros* исто што и Arial у Windows, док је *termes* исто што и Times New Roman. Они нису потпуно идентични, већ довољни слични, толико да морате бити заиста оштрог ока да бисте приметили разлику.

Фонтови које користи Windows нису *слободан софтвер* (уствари, скоро све у Windows систему није *слободан софтвер*), тако да не смеју бити део ConT_EXt дистрибуције. Међутим, ако је ConT_EXt инсталиран на Windows систему, онда су ови фонтови већ инсталирани и могу да се користе као и било који други фонт инсталиран на систему који извршава ConT_EXt. Ипак, у овом уводу се нећу бавити начином употребе фонтова који су већ инсталирани на систему. Помоћ у вези тога можете пронаћи на [ConT_EXt викију](#).

6.2 Особине фонта

6.2.1 Фонтови, стилови и стилске варијанте

Терминологија у вези фонтова је донекле збуњујућа, јер је оно што се понекада назива фонт уствари *фамилија фонтова*. Она се састоји од различитих стилова и варијанти које деле основни дизајн. Нећу се упуштати у расправу о томе која терминологија је тачнија; интересује ме само да да разјасним терминологију коју користи ConT_EXt. Ту се прави разлика између фонтова, стилова и варијанти (или алтернатива) сваког стила. *Фонтови* укључени у ConT_EXt дистрибуцију (то су уствари *фамилије фонтова*) су они које смо видели у претходном одељку. Сада ћемо погледати у *стилове* и *алтернативе*.

Стилови фонтова

Доналд Е. Кнут је за T_EX дизајнирао *Computer Modern* фонт и дао му три различита *стила* која се називају *roman*, *sans serif* и *teletype*. *Roman* стил је дизајн у коме карактери имају декоративне украсе који су у типолошкој терминологији познати под именом *стопице* (*серифи*), па је из тог разлога овај стил фонта познат и под називом *serif*. Овај стил се узима као *нормални* или подразумевани стил. *Sans serif* стил, као што му име наговештава, нема ове украсе, па је стога једноставнији, сведенији фонт, понекада познат под другим именима, нпр. линеарни, на шпанском, *paloseco*; овај фонт може бити главни фонт у документу, али је такође погодан за употребу у одређеним фрагментима текста чији је главни фонт *roman* стила, као на пример, у насловима или заглављима страница. Коначно, *teletype* стил је укључен у *Computer Roman* јер је он био дизајниран за писање књига о компјутерском програмирању, које имају дугачке одељке компјутерског *кода*, а он се на штампаним материјалима традиционално представља стилем једнаке ширине који имитира компјутерске терминале и старе писаће машине.

Уз ова три *стила* фонтова би могао да се дода и четврти стил намењен за математичке фрагменте. Али пошто T_EX аутоматски користи овај стил када уђе у математички режим, и не поседује команде којима се експлицитно укључује или искључује, нити поседује *варијанте* или алтернативе осталих стилова, није уобичајено да се сматра за *стил* у ужем смислу.

ConT_EXt нуди команде за два могућа стила: рукописни и калиграфски. Нисам потпуно сигуран шта је разлика између њих јер, с једне стране, ниједан од фонтова укључених у ConT_EXt дистрибуцију нема ове стилове у свом дизајну, а са друге стране, према мом мишљењу, калиграфско писање је такође рукописно. Ове команде које ConT_EXt нуди за укључивање оваквих стилова, ако се користе са фонтом који их не имплементира, неће изазвати никакву грешку приликом компајлирања: једноставно се ништа неће догодити.

Алтернативне форме фонта

Сваки *стил* поседује више алтернативних форми, а ConT_EXt их тако и назива, (*алтернативе*):

- Регуларни или нормални („tf“, од *typeface*).
- Црни слог („bf“, од *boldface*).
- Курзив („it“ од *italic*)

- Црни курзив („bi“ од *bold italic*)
- Коси („sl“ од *slanted*)
- Црни коси („bs“ од *bold slanted*)
- Капитал („sc“ од *small caps*)
- Средњовековни („os“ од *old style*)

Ове *алтернативе* су, као што им име наговештава, узајамно искључиве: када се једна укључи, остале се искључују. То је разлог што ConTeXt нуди команде за укључивање алтернатива, а команде за искључивање не постоје; јер кад укључимо једну алтернативу, искључујемо ону која је била у употреби до тад; тако на пример, ако пишемо курзивом и укључимо црни слог, онде се курзив искључује. Ако желимо да користимо црни слог и курзив истовремено, не морамо да их укључујемо појединачно, већ само треба да укључимо алтернативу која има оба („bi“).



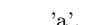


С друге стране, треба имати на уму да мада ConTeXt претпоставља да сваки фонт поседује ове алтернативе, па зато и обезбеђује команде које их укључују, да би оне функционисале и давале видљиви резултат у финалном документу, потребно је и да дизајн фонта поседује одређене форме за сваки стил и алтернативу.






Уствари, многи фонтови не праве разлику у дизајну између косих слова и курзива, нити обезбеђују специјалне облике за капитал.


Разлика између курзива и косих слова

Сличност типографске функције коју имају курзив и коса слова је узрок што многи људи мешају ове две алтернативе. Коса слова се добијају благом ротацијом основног облика. Али курзив је – барем у неким фонтовима – другачији дизајн у којем слова *изгледају* искошена јер су тако исцртана; али у стварности нема стварног искошења. Ово се може видети у наредном примеру, у којем смо исписали исту реч три пута у истој величини, довољној да се лако виде разлике. У првој верзији је употребљена регуларна форма, у другој искошена, а у трећој курзив:

italics – italics – italics

1.     

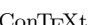


2.     


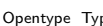

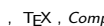


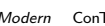



1. 

2. 

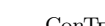

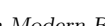
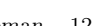







ConTeXt    6.5.1.






6.2.2






ConTeXt    12 .






ConTeXt          






6.3

ConTeXt    12 .        

- , 6.1.
- , () . ConT_EXt , `\setupbodyfont` 4 12, 14.4 17.3. 12 .
`\setupbodyfont`, ; , . `\setupbodyfont` , .
(10pt, 11pt, 12pt, .) . , : big, small, script, x, scriptscript xx. , `\setupbodyfont`
12 , „big”.
- , , roman (), (sans serif), (handwritten) (calligraphic) . `\setupbodyfont` . 6.2:

| | |
|--------------|------------------------------|
| | |
| Roman | rm, roman, serif, regular |
| Sans Serif | ss, sans, support, sansserif |
| Monospaced | tt, modo, type, teletype |
| Handwritten | hw, handwritten |
| Calligraphic | cg, calligraphic |

6.2 `\setupbodyfont`

6.3.1

, . , ; , ConT_EXt ConT_EXt. `\showbodyfont`, .
`\showbodyfont` :
`\showbodyfont []`
`\setupbodyfont`. , `\showbodyfont[schola, 8pt]` , schola 8 :

| [schola] [schola,8pt] | | | | | | | | | | \mr : Ag | | | |
|-----------------------|-----|-----|-----|-----|-----------|-----------|-----------|------|-------|----------|------|------|-----------|
| | \tf | \sc | \sl | \it | \bf | \bs | \bi | \tfx | \tfxx | \tfa | \tfb | \tfc | \tfd |
| \rm | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag |
| \ss | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag |
| \tt | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag |

, , `\showbodyfont`.
, `\showfont []`.

6.4

6.4.1 `\setupbodyfont` `\switchtobodyfont`

, , ConT_EXt : `\setupbodyfont`. . , , .
`\switchtobodyfont` `\setupbodyfont`. (,), , . `\setupbodyfont` () ; , (). ,
`\switchtobodyfont` , .
— .:
1. , `\setupbodyfont` , `\switchtobodyfont` , , .
2. `\switchtobodyfont` , `\setupbodyfont` , , , .
, , , , , .
`\setupbodyfont` ; , (Latin Modern Roman) . `\switchtobodyfont` , , , , , .

6.4.2 ,

`\switchtobodyfont`, ConT_EXt , . , ConT_EXt , , , `\dontleavehmode` .

| | |
|--------------|---|
| Roman | <code>\rm, \roman, \serif, \regular</code> |
| Sans Serif | <code>\ss, \sans, \support, \sansserif</code> |
| Monospaced | <code>\tt, \mono, \teletype,</code> |
| Handwritten | <code>\hw, \handwritten,</code> |
| Calligraphic | <code>\cf, \calligraphic</code> |
| 6.3 | |
| Normal | <code>\tf, \normal</code> |
| Italic | <code>\it, \italic</code> |
| Bold | <code>\bf, \bold</code> |
| Bold-italic | <code>\bi, \bolditalic, \italicbold</code> |
| Slanted | <code>\sl, \slanted</code> |
| Bold-slanted | <code>\bs, \boldslanted, \slantedbold</code> |
| Small caps | <code>\sc, \smallcaps</code> |
| Medieval | <code>\os, \mediaeval</code> |
| 6.4 | |

6.3 ; 6.4 .

, , , , , *thought* , , .

I thought a `{\it thought}` but
the `{\it thought}` I thought wasn't
the `{\it thought}` I thought I thought.
If the `{\it thought}` I thought I thought
had been the `{\it thought}` I thought

I wouldn't have thought so much!

I thought a *thought*, but the *thought* I thought, wasn't the *thought*
I thought I thought. If the *thought* I thought I thought had been
the *thought* I thought I wouldn't have thought so much!

`(\tf, \it, \bf, .)` . a, b, c d , 1.2, 1.2² (= 1.44), 1.2³ (= 1.728) 1.2⁴ (= 2.42). :

`\tf , \tfa , \tfb , \tfc , \tfd`

`, , , ,`

x xx 0,8 and 0,6: respectively:

`\tf , \tfx , \tfxx`

`, ,`

'x' 'xx' `\tf` , `\tfx` `\tx`, `\tfxx` `\txx`.

. ConT_EXt 2013. (Mark II) 'x', ; .

, `\showbodyfont` (6.3.1). , , .

`\showbodyfont:`

| [modern-designsize] | | | | | | | | | | | | \mr : Ag | |
|---------------------|------------------|------------------|------------------|------------------|------------------|------------------|------------------|-------------------|--------------------|-------------------|-------------------|-------------------|-------------------|
| | <code>\tf</code> | <code>\sc</code> | <code>\sl</code> | <code>\it</code> | <code>\bf</code> | <code>\bs</code> | <code>\bi</code> | <code>\tfx</code> | <code>\tfxx</code> | <code>\tfa</code> | <code>\tfb</code> | <code>\tfc</code> | <code>\tfd</code> |
| <code>\rm</code> | Ag | AG | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag |
| <code>\ss</code> | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag |
| <code>\tt</code> | Ag | AG | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag | Ag |

, (`\rm, \ss \tt`). , , (`\tf, \sc, \sl, \it, \bf, \bs \bi`), , .

, , , .

`\definebodyfontenvironment` :

`\definebodyfontenvironment[] []`
`\definebodyfontenvironment[default] []`

`\setupbodyfont` `\switchtobodyfont.` :

`\definebodyfontenvironment[10pt] [a=12pt,b=14pt,c=2, d=3]`

```

10, 'a' 12, 'b' 14, 'c' 2, 'd' 3. a b , c d .

\definebodyfontenvironment „default”, , ., :

\definebodyfontenvironment[default][a=1.3,b=1.6,c=2.5,d=4]

, a 1,3, b 1,6, c 2 d 4.

xx, x, a, b, c d, \definebodyfontenvironment „big”, „small”, „script” „scriptscript”.
\setupbodyfont \switchtobodyfont. , ( ) :

• \smallbold
• \smallslanted
• \smallboldslanted
• \smallslantedbold
• \smallbolditalic
• \smallitalicbold
• \smallbodyfont
• \bigbodyfont

, \showbodyfontenvironment[. , , modern , :
```

| [modern] | | | | | | | |
|----------|--------|--------------|--------|--------|--------|--------|----------------|
| text | script | scriptscript | x | xx | small | big | interlinespace |
| 10pt | 7pt | 5pt | 8pt | 6pt | 8pt | 12pt | |
| 11pt | 8pt | 6pt | 9pt | 7pt | 9pt | 12pt | |
| 12pt | 9pt | 7pt | 10pt | 8pt | 10pt | 14.4pt | |
| 14.4pt | 11pt | 9pt | 12pt | 10pt | 12pt | 17.3pt | |
| 17.3pt | 12pt | 10pt | 14.4pt | 12pt | 14.4pt | 20.7pt | |
| 20.7pt | 14.4pt | 12pt | 17.3pt | 14.4pt | 17.3pt | 20.7pt | |
| 4pt | 4pt | 4pt | 4pt | 4pt | 4pt | 6pt | |
| 5pt | 5pt | 5pt | 5pt | 5pt | 5pt | 7pt | |
| 6pt | 5pt | 5pt | 5pt | 5pt | 5pt | 8pt | |
| 7pt | 6pt | 5pt | 6pt | 5pt | 5pt | 9pt | |
| 8pt | 6pt | 5pt | 6pt | 5pt | 6pt | 10pt | |
| 9pt | 7pt | 5pt | 7pt | 5pt | 7pt | 11pt | |

6.4.3

```

,

, ., ConTEXt :

1. , .

2. \switchtobodyfont.

:

• \definebodyfontswitch: ., \osam ( \viii1) 8 pt, :

\definebodyfontswitch[osam][8pt] \definebodyfontswitch[viii][8pt]

• \definefontstyle: \setupbodyfont \switchtobodyfont ; , sans serif (. „linear”)

\definefontstyle[linear][ss]

\definefontstyle , :

\definefontstyle[linear, tehn, bezstopica][ss]

• \definealternativestyle: . style .,

\definealternativestyle[strong][\bf] []
```

¹ Упamtите да за разлику од командних симбола, имена команди у систему ConT_EXt смеју да се састоје само од слова.



`\strong` „strong” style . „bold”, ConT_EXt , HTML, „strong”
`\definealternativestyle.` , ; ConT_EXt „, \cap, ” (??)

6.5

, :

6.5.1 ,

, , .
 , . ConT_EXt , . `\em emphasis.` `\it \sl`, , `\em` ; , , ConT_EXt `\em \it`
`\sl.` , ConT_EXt ; `\em` , ConT_EXt . , , — — `\em`, — .
 :

```
{\em
\em Thelymitra variegata}
}
```

Thelymitra
 variegata .

`\em (,), \em „Thelymitra variegata” .`
`\em , , , \em \bs. , \bf, .`
`\em , \setupbodyfontenvironment[default][em=italic].`

6.5.2

(). , (). . , , .
 , , , , , , .
 . ConT_EXt `\cap \Cap`; 10.2.1.

6.6

ConT_EXt , , .

6.6.1

ConT_EXt „color” , , :

```
\setuphead
[chapter]
[color=blue]
```

```
,,, , , , ( ) .
, , , , , , ,
```

```
\color[red]{ }
```

```
.
\definehighlight[vazno][color=red]
\vazno{ }
```

6.6.2

, (), `\setupbackgrounds \setupcolors.` ,

```
\setupbackgrounds
[page]
[background=color,backgroundcolor=blue]
```

```
. „backgroundcolor” .
```

```
( ) \setupcolors, „textcolor” . . :
\setupcolors[textcolor=red]
```

6.6.3

```
\color[]{}

\startcolor[] ... \stopcolor

. . . , \colored. :

\colored[r=0.1, g=0.8, b=0.8]
{} .
```

6.6.4

ConT_EXt 6.5.¹

| | | | |
|---------|------------|---------------|------------|
| black | | | |
| white | | | |
| gray | lightgray | middlegray | darkgray |
| red | lightred | midddlered | darkred |
| green | lightgreen | middlegreen | darkgreen |
| blue | lightblue | middleblue | darkblue |
| cyan | | middlecyan | darkcyan |
| magenta | | middlemagenta | darmagenta |
| yellow | | middleyellow | darkyellow |

6.5 ConT_EXt

```
,
\usecolors[]

• „crayola”, 235 .
• „dem”, 91 .
• „ema”, 540 Emacs.
• „rainbow”, 91 .
• „ral”, 213 Deutsches Institut für Gütesicherung und Kennzeichnung ( ).
• „rgb”, 223 .
• „solarized”, 16 solarized .
• „svg”, 147 .
• „x11”, 450 X11.
• „xwi”, 124 .

„context/base/mkiv” „colo-imp-.mkiv”. . , .






























, \showcolor[] . , (\usecolors[]), \color \startcolor . , :

\usecolors[xwi]
\color[darkgoldenrod]{} }
```


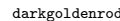

¹ Ова листа може да се пронађе у референтном упутству и на ConT_EXt викију, али сам прилично сигуран да није комплетна јер у овом документу, на пример, без потребе за учитавањем било какве додатне боје, користимо „наранџасту” – која није у [табели 6.5](#) – за наслове одељака.

6.6.5

`\showcolor` , , , ConT_EXt. , `\showcolor` , 6.6.4, `\showcolor[solarized]` , 16 solarized :

| | | | | | | |
|---|---|-------|-------|-------|-------|---------|
|  |  | 0.561 | 0.514 | 0.580 | 0.588 | base0 |
|  |  | 0.460 | 0.396 | 0.482 | 0.514 | base00 |
|  |  | 0.409 | 0.345 | 0.431 | 0.459 | base01 |
|  |  | 0.162 | 0.027 | 0.212 | 0.259 | base02 |
|  |  | 0.123 | 0.000 | 0.169 | 0.212 | base03 |
|  |  | 0.615 | 0.576 | 0.631 | 0.631 | base1 |
|  |  | 0.909 | 0.933 | 0.910 | 0.835 | base2 |
|  |  | 0.965 | 0.992 | 0.965 | 0.890 | base3 |
|  |  | 0.457 | 0.149 | 0.545 | 0.824 | blue |
|  |  | 0.487 | 0.165 | 0.631 | 0.596 | cyan |
|  |  | 0.510 | 0.522 | 0.600 | 0.000 | green |
|  |  | 0.429 | 0.827 | 0.212 | 0.510 | magenta |
|  |  | 0.422 | 0.796 | 0.294 | 0.086 | orange |
|  |  | 0.395 | 0.863 | 0.196 | 0.184 | red |
|  | | 0.473 | 0.424 | 0.443 | 0.769 | violet |
| | | 0.530 | 0.710 | 0.537 | 0.000 | yellow |

`rgb` , `\showcolorcomponents[]` , . , `\showcolorcomponents[darkgoldenrod]` :

| color | name | transparency | specification |
|---|---------------|--------------|-------------------------|
|  | white | | |
|  | black | | |
|  | darkgoldenrod | | r=0.720,g=0.530,b=0.040 |

6.6.6

`\definecolor` , . . , :

`\definecolor[][]`

„ ” „ ” .

`\definecolor` . , :

`\definecolor[][]`

:

- **RGB** : RGB ; , , : ('r' *red*), ('g' *green*) ('b' *blue*). 0 1.

`\definecolor[limeta 1][r=0.75, g=1, b=0]: „limeta 1”.`

- **Hex** : RGB , , , .

`\definecolor[limeta 2][x=BFFF00]: „limeta 2”.`

- **CMYK** : „ ” : ('c'), ('m'), ('y', *yellow*) ('k', *key*). 0 1.

`\definecolor[limeta 3][c=0.25, m=0, y=1, k=0]: „limeta 3”.`

- **HSL/HSV**: ('h', *hue*), ('s' *saturation*) ('l' 'v', *value*). 0 360; 0 1.

`\definecolor[limeta 4][h=75, s=1, v=1]: „limeta 4”`

- **HWB** : HWB CSS4 ('h', *hue*), ('w', *whiteness*) ('b', *blackness*). 0 360, 0 1.

`\definecolor[Azure][h=75, w=0.2, b=0.7] „Azure”.`

- : ('s', *scale*) . 0 1. :

`\definecolor[svetlo siva][s=0.65] „svetlo siva”.`

,

`\definecolor[maincolour][0.6(orange)]`

7

```

: 7.1 ; 7.2 ; 7.3 ; 7.4 ; 7.4.1 \setuphead \setupheads; 7.4.2 ; 7.4.3 ( ); 7.4.4 ; 7.4.5 ; 7.4.6
; 7.4.7 ; 7.4.8 \setuphead ; 7.5 ; 7.6 ;

```

7.1

[illegible]ConT_FXt

```

    . "
    ,
    ConTExT "section" ( ).
    , ConTExT

```

”() :

- „... *per se* ...“
- ConT_{FX}t

- " " :
 " " " " :
 " " " " :

$$= () .$$

—

ConTeXt, ., ConTeXt, .

7.2

ConTeXt . . . / .

ConT_FXt 7.1.

| | | |
|-----|-------------------|-------------------|
| 1 | \part | — |
| 2 | \chapter | \title |
| 3 | \section | \subject |
| 4 | \subsection | \subsubject |
| 5 | \subsubsection | \subsubsubject |
| 6 | \subsubsubsection | \subsubsubsubject |
| ... | ... | ... |

7.1 ConT_FXt

•

- 7.1 . (\startchapter ... \stopchapter,) .
- 6 ., 12 : \subsubsubsection \subsubsubsubsection, , \subsubsubsubsubsubsubsubsubsection, \subsubsubsubsubsubsubsubsubsubject.

() ! , . , , . ; , .

„sub“ , : „subsubsubsubsection“ „sub“-! , , (subsubsection) (7.5).

- $(\backslash part)$, , (), .
 ConTtXt , $\backslash part$; , , , .

- \bullet " "

7.3

```

, ( 7.5), (, , „section”):

\section [] {}
\section []
\startsection [] [] ... \stopsection

, . , Mark IV .

• , „”, , , ( 9.2).

• ConTeXt :

3.3.1 3.4 ConTeXt , . , , , . ConTeXt TeX,

(=) :

- reference: .
- title: .
- list: .
- marking: .
- bookmark: bookmark PDF .
- ownnumber: ; .

, „list”, „marking” „bookmark” „title”. , , ; \nomarking \nolist ( ). , (
„title”) , , , ConTeXt : „list”, „marking” „bookmark”. , , .

, „ ” „test”, „ ” „ ”.

\chapter
[
  title={ },
  reference={test},
  marking={ }
]

\start . , , , ConTeXt . Mark IV ; . , XML EPUB ., XML .

\start . \structureuservariable.

ConTeXt , , , , ConTeXt .

```

7.4

7.4.1 \setuphead \setupheads

```

ConTeXt ( ) , . \setuphead :

\setuphead[] []

• ( ) . :

- (part, chapter, title, .), , . „section-”, . „section-1” „part”, „section-2”
  „chapter”, .

- . , 7.5.

• (=). ( ) . , .

\setuphead . : , . :

• , ; : . , , „chapter” ( 2) „title”; „title”, „chapter”.

• , , . , : , , , . , .

```



```

\setuphead, ConTeXt \setupheads . , ConTeXt . , . , „style”, , . , ,
, „number” „color”. , \setupheads[color=blue] .
\setupheads ( ) \setuphead.
: , ConTeXt , : \start .
\setupheads, \setuphead. , , , „insidection” .

```

7.4.2



```

, . ConTeXt , . :
• , . , „part” „placehead” „yes”, „no” „hidden”, „empty” „section”. .
, , :
\setuphead
[part]
[placehead=yes]
, . list marking . \nolist \nomarking . :
\chapter{ \nomarking{19. 21. }}
„ ...”.
• . (part, chapter, section, subsection...), (title, subject, subsubject). , „number” „incrementnumber”
„yes” „no”. yes no.
? , ; (incrementnumber), (number). incrementnumber=yes number=no, (), . , . , 8.1.7.
• . , . , „”, „” parts. \setuphead \setuplabeltext. . , „”, :
\setuplabeltext
[chapter=-]
„~” . , . ( ) ; , „1” „1”.

```

7.4.3 ()

```

(part, chapter, section...) , „number” „incrementnumber” \setuphead.
, „ownnumber” „yes”. „ownnumber=yes” . :
• , . : \chapter{13}{ } 13.
• ConTeXt (\ [] \start []), „ownnumber”. : \chapter[title= , ownnumber=13], 13.
ConTeXt , ; , , . ConTeXt , :
• '1' .
• '0' .
, 1 , , . 0; .
, \setupheadnumber :
\setupheadnumber[] [ ]
. , 1; 10, 11.
; , , . , \setupheadnumber[section][+5] 5 ; \setupheadnumber[chapter][14, +5] 15
(14+1), 20 (15+5), 25, .
, . : , , , . „1.3.2.4”.
:
• conversion: . , :
– : : 1, 2, 3, ... n, N numbers.
– . :

```

```

*      : I, R, Romannumerals.
*      : i, r, Romannumerals.
*      : KR, RK.

-      :

*      : A, Character
*      : a, character
*      : AK, KA

-      : '3' 'Three'.

*      : Words.
*      : words.

-      : ConTeXt , . ConTeXt : set 0, set 1, set 2 set 3. . set 0
set 1 9, set 2 set 3 12:

Set 0: • - * ▷ ° ○ □ ✓
Set 1: * ** *** † ‡ †† * ** ***
Set 2: * † ‡ ** †† ‡‡ *** ††† ††† **** †††† †††††
Set 3: * ** *** † ‡ ††† ¶ ¶¶ ¶¶¶ § §§ §§§

• sectionsegments: . . (part=1, chapter=2, section=3, .), (part, chapter, section,
.). , „sectionsegments=2:3” . „sectionsegments=chapter:section”. , „optionsegments”
Initial Level:all, Initial Level:*. , „sectionsegments=3:*” 3 (section).

, , ; , ; , . :

\setuphead[part][conversion=I]
\setuphead[chapter][conversion=n, sectionsegments=2]
\setuphead[section][conversion=n, sectionsegments=2:3]
\setuphead[subsection][conversion=n, sectionsegments=2:4]
\setuphead[subsubsection][conversion=A, sectionsegments=5]

```

7.4.4

```

:
• „style”, „numberstyle” „textstyle” , . ; , (roman, sans serif typewriter), (italic, bold, slanted...) . , ( „bold” ), („bf”), (\bf, ). , , . , , .
• „color”, „numbercolor” „textcolor” , . ConTEXt , . .
, . : „command”, „numbercommand”, „textcommand”, „deepnumbercommand” „deeptextcommand”. :
• command , . ConTEXt , .
• numbercommand „command”, .
• textcommand „command”, .
, , , , , „command”. :
\define[2]\AlignSection
{ \framed[frame=on, width=, align=flushright]{#1\#2} }
\setuphead
[section]
[command=\AlignSection]
„command” „style”, , , „textstyle=\em”, „textcommand=\WORD”, \WORD ( ) , . : \WORD{ \em
} . , „textcommand” „numbercommand” „deeptextcommand” „deepnumbercommand”. { \em \WORD{
} }”

```

7.4.5

```
„alternative” :      ( )      . , .
„alternative”:
```

- `text:` . `LATEX` `\paragraph` `\subparagraph`.
- `paragraph:` .
- `normal:` `ConTEXt` . „`paragraph`”.
- `middle:` , . , .
„`alternative=middle`”, „`align`” . „`left`”, „`middle`” „`flushright`”. , .
- `marginintext:` () .
„`alternative`”:
- `margin/inmargin:` . . „`margin`” „`inmargin`”.
- `reverse:` , , , .
- `top/bottom:` , .

7.4.6



```
( „before” ( „after”). : „before=\blank” . , „before={\blank[3*big]}”.
„before=\hairline, after=\hairline”, .
„before” „after” „commandbefore” „commandafter”. , ,
, „page” , „yes” , „left” , „right” , „no” . , „chapter” „continue=no”,
, .
ConTEXt, : , . „continue”, , : „continue=yes”, . „continue=no”
(\start ... \stop), „insidection” . , , („insidection=\placecontent”)
```

7.4.7

- , `\setuphead` :
- . „`interlinespace`” `\defineinterlinespace` `\setupinterlinespace`.
- . „`align`” : „`flushleft`” (), „`flushright`” (), „`middle`” (), „`inner`” () „`outer`” () .
- . „`margin`” .
- . „`indentnext`” („yes”, „no” „auto”) . () .
- . , , . „`width`” . „`numberwidth`” „`textwidth`” .
- . „`distance`” „`textdistance`” .
- . „`header`” „`footer`”.



7.4.8 `\setuphead`

• , `\setuphead` . , .

7.5

```
\definehead :
\definehead[] [] []
```

- .
- .
: , , , .
- . `\setuphead`.
• `\setuphead`, , `ConTEXt` , .

7.6

$$, , , \dots, ; \quad , \quad , \quad , \quad - \quad , \quad , \quad , \quad \vdots$$

-
- Figure 1 consists of four horizontal timelines labeled (a), (b), (c), and (d). Each timeline has a vertical line on the left representing the start of the trial. Timeline (a) shows a stimulus (S) and a response (R) occurring at the same time. Timeline (b) shows S occurring before R. Timeline (c) shows S occurring after R. Timeline (d) shows S occurring before R, with a longer interval between them than in (b).

7.2.

| | | | |
|--------------------------------|-----------------|------------------|-------------------------------|
| <code>\startfrontmatter</code> | <code>[]</code> | <code>...</code> | <code>\stopfrontmatter</code> |
| <code>\startbodymatter</code> | <code>[]</code> | <code>...</code> | <code>\stopbodymatter</code> |
| <code>\startappendices</code> | <code>[]</code> | <code>...</code> | <code>\stopappendices</code> |
| <code>\startbackmatter</code> | <code>[]</code> | <code>...</code> | <code>\stopbackmatter</code> |

7.2

```

: „page”, „before”, „after” „number”, \setuphead ( 7.4), „number=no”
.
. \setupsectionblock :

```

`\setupsectionblock[] []`

```
frontpart, bodypart, appendix backpart, : „page”, „number”, „before” „after”. , frontmatter ( )
:
```

```
\setupsectionblock
[frontpart]
[
  before={\setuppagenumbering[conversion=Romannumerals]}
]
```

ConT_EXt :

- .
 - :
 - frontmatter backmatter .
 - bodymatter .
 - appendices .
- `\definesectionblock.`

```

: 8.1 ; 8.1.1 ; 8.1.2 ; 8.1.3 ; 8.1.4 : criterium; 8.1.5 : alternative; 8.1.6 ; 8.1.7 ; ; ;
; ; 8.2 , ; 8.2.1 ConTeXt; 8.2.2 , ; 8.2.3 ; 8.3 ; 8.3.1 ; ; 8.3.2 ; 8.3.3 ;
. , . ( ) ”. , ( , , ), .

```

9.2.

8.1

8.1.1

„ , , () . , , , () , ; , .

ConTeXt . :

- .
- .
- , .
- .

„ , , 9.3.

ConTeXt , , . ConTeXt ; . , , : – – . ():
ConTeXt . , , – – . , ; , .

8.1.2

```

( , , .) \completecontent \placecontent. ; , .
, \completecontent:
• , , ”.
ConTeXt \title ( 7.2). \completecontent (\chapter) (\title). , , .
• (, \title) ; .
• .
, (\completecontent). , , 8.1.3, , ( \placecontent), \completecontent ,
; . , „criterium=all”. , 8.1.3.
, \setupheadtext :
\setupheadtext [] [=]
ConTeXt ( 10.5), („content” ) .
\setupheadtext [en] [content=Contents]
\completecontent „Contents” „Table of Contents”.
, \completecontent \placecontent, ( ).

```

8.1.3

„ , \placecontent :

```
\placecontent[]
, , \setupcombinedlist ( ). , :
\setupcombinedlist[content][list={chapter,section}]
.
. , . :
• \placecontent , (part, chapter section, ) .
, , ,
\setupcombinedlist[content][list={chapter, section, subsection, ...}]
• , , , , ( ) , .
, \placecontent, ConTEXt Mark IV \start, \stop . , \placecontent :
• ( ) . , .
• . , , ; , .
, , \placecontent , , .
. : :
▪ 1
- 1.1
- 1.2
* 1.2.1
* 1.2.2
* 1.2.3
- 1.3
- 1.4
▪ 2
: \placecontent 1, , \completecontent, . 1, 1.1, ; 1.2, . , 1.1 1.2, . , .
,, ., criterium .
\placecontent , , () ConTEXt . criterium ; alternative, .
```

8.1.4 :
criterium

```
\placecontent . criterium . , :
• all: , .
• previous: ( ) \placecontent. .
• part, chapter, section, subsection...: .
• component: ( 4.6), \placecontent \completecontent.
```

8.1.5 : alternative

alternative . 8.1.

| alternative | | |
|-------------|-------|---|
| a | - - | |
| b | - - - | |
| c | - - - | |
| d | - - | |
| e | | |
| f | | , |
| g | | |

8.1

alternative (,), . , , , ConT_EXt .
, , .

8.1.6

```

alternative \placecontent \completecontent , . , . \setuplist :
\setuplist[] []
. part, chapter, section, . , . 54, , , ; , .
, , , , alternative, : , . , :
• ( ) : ( 'a' 'b' 'c' 'd'), headnumber=no pagenumber=no , (headnumber) (pagenumber) .
• : ( ) : , . style color , numberstyle, textstyle pagestyle ( ) numbercolor,
textcolor pagecolor ( ) .
, . command, numbercommand, pagecommand textcommand. ConTEXt , . ,
command, textcommand , pagecommand . , , ( ) :
\setuplist[section][textcommand=\Cap]
• : before after (before) (after) .
• : margin .
• : . interaction (interaction=no), , (interaction=number interaction=sectionnumber),
(interaction=text interaction=title) (interaction=page interaction=pagenumber).
• :
– width: . , fit .
– symbol: . : one, two three. none .
– numberalign: ; left, right, middle, flushright, flushleft.
, . , . , (\placecontent \completecontent) . :
\start
\setupinterlinespace[small]
\placecontent
\stop

```

8.1.7

```

(\placecontent \completecontent), . , , . . :
• .
• .
• .
• .
• .
• .
ConTEXt , , ( 7.4.2) ( incrementnumber) \setuphead . incrementnumber=yes
number=no , .
, – , title – , incrementnumber , yes, , , , \setupcombinedlist:
\setuphead
[title]
[incrementnumber=yes]
\setupcombinedlist
[content]
[list={chapter, title, section, subsection, subsubsection}]
, , \setuplist ; :
\setuplist[title][style=bold]
: ( , title). , .

```



```

•
    ( ), .
    , \writetolist :
\writetolist[] []{}{}

•
    : chapter, section, subsection, etc.
• , , . , ; .
    ConTeXt ( 3.6) \setuplist, . , .
•
    , .
•
    , , , . ; , \writetolist. , , , :
\subject{Bibliography}
\writetolist[section]{}{Bibliography}

    section, subject, , (section).

    \writebetweenlist , , , :
\writebetweenlist[section]{\hrule}

•
    , , list \setupcombinedlist, , .
    , , , , title chapter, subject section, . , .
    , , , , . :
\definehead[MySubsection][subsection]
\section{ }
\subsection{ }
\MySubsection{ }
\subsection{ }

    MySubsection, , , MySubsection.

•
    , :
•
    ({} ) \ [], \start [], list ( 7.3).
•
    , \nolist: . :
\chapter
[title={\nolist{ }
}]
    , „ ”, „... ”.
    : \nolist ConTeXt . , \nolist .

```

8.2 ,

ConTeXt, , , . ConTeXt , . ConTeXt „content” : `\placecontent`
`\completecontent`.

8.2.1 ConTeXt

ConTeXt :


```

1. .
2. .
3. .

; , , , \place , \placetable, \placefigure, .
, ConTeXt , , . , , chapter, , section; ( table) ( figure). ConTeXt .
, , : , . incrementnumber=yes, , .
( ConTeXt ) \definelist .
\definelist[] []
:
• ConTeXt, ( 13.5), , , \place , : \placefigure , .
• \writetolist[], 8.1.7. \writebetweenlist. .
, \setuplist, \placelist \completelist. ; . \placelist \completelist
\placecontent \completecontent ( 8.1.2 8.1.3).
,
\placelist[section]
, , . \setuplist set interaction=no. : ( , , .) .
:
\placelist[] []
\setuplist[] []
\setuplist 8.1.6, \placelist \placecontent ( 8.1.3).

8.2.2 ,
, ConTeXt \placefigure, \placelist[figure]. ( \completecontent)
\completelist[figure]. ConTeXt: („table”), („graphic”), („intermezzo”) („chemical”),
, ConTeXt : (\placelistoffigures, \placelistoftables, \placelistofgraphics, \placelistofintermezzi
\placelistofchemicals), : (\completelistoffigures, \completelistoftables, \completelistofgraphics,
\completelistofintermezzi \completelistofchemicals), \completecontent.
, ( 13.5) \placelistof<> \completelistof<>.
\definelist, \placelist[] \completelist[].

8.2.3
, , . ConTeXt „content”, \definecombinedlist :
\definecombinedlist[] [] []
• : .
• : , .
• : . , , , . ( ) criterium ( 8.1.4 8.1.3) alternative ( 8.1.5 ).
\definecombinedlist \place , : . , ,
definecombinedlist[TOC]
\placeTOC;
definecombinedlist[content]
\placecontent

```

```
, \placecontent!      ? : ConTeXt      :
\definecombinedlist
[content]
[part, chapter, section, subsection,
subsubsection, subsubsubsection,
subsubsubsubsection]

, ( ) \setupcombinedlist criterium ( 8.1.4 8.1.3) alternative ( 8.1.5 ), list .
ConTeXt ( 3.6) \setupcombinedlist list, ( , , ). .
```

8.3

8.3.1

```
. .
, , . . . , , . . .
:
1. , . . .
2. . , , , ConTeXt , . .
3. , . ConTeXt : \placeindex.
.
. , . . : , . .
, , , \index :
\index[Alphabetical]{ }
where , , . , , ConTeXt .
. , , \TeX. \index{\backslash TeX} , , 't' 'TeX', , \index[tex]{\backslash TeX}.
. , , ; , , ” ” ” ” ” ”
, , .
( ), '+'. :
\index{ 1+ 2}
\index{ 1+ 2+ 3}
2 1. 3 2, 1.
My \index{}, \index{+} .
\index{+} .
:
• \index . :
— , ConTeXt , , .
— , . . .
• .
• \index , ” ” ” ” ” ”
• : . . . . .
, . . , , .
.
.
```

```

\index, \seeindex :
\seeindex[] {1} {2}
1 ; 2 . , :
\seeindex{ }{}
\seeindex '+' .
•
, \placeindex \completindex. \index , \index. , , .
\placeindex \completeindex \content \completecontent ( 8.1.2): \placeindex , \completeindex
” , .

```

8.3.2

```

ConTeXt „”; :
\setupregister[index] []
:
• .:
– . alternative=A .
– . textstyle, textcolor, textcommand deeptextcommand , pagestyle, pagecolor pagecommand,
. pagenumber=no ( ).
– distance ; .
style, textstyle, pagestyle, color, textcolor, pagecolor . command, pagecommand, textcommand
deeptextcommand, 7.4.4, .
• , (before) (after), (n), (balance), (align), .

```

8.3.3

```

; . , , . . ; , , .
\defineregister :
\defineregister[] []
, .
\setupregister[] []
\ , \index. \see .
: ConTeXt :
\defineregister[macro]
\macro. ConTeXt , \placemacro \completemacro .
\ , \placeIndexName \completeIndexName . \placeIndexName \placeregister[], \completeIndexName
\completeregister[].

```

```

: 9.1 ; 9.2 ; 9.2.1 ; 9.2.2 ; ; \ref; ; 9.2.3 ; 9.3 ; 9.3.1 ; 9.3.2 ; 9.4 ;
9.4.1 , ; 9.4.2 ; 9.5 PDF ;

```

9.1

```

:
• , , , , .
• , . , , , , .
, , . ( , , „3.2”, „7”), . „ ” ” , , .
„ ” „ ” , „ ” .
, , , , , .
:
• , : , , , , .
• , , , , , .
, , ( , , ), .
, , . ConTeXt, .

```

9.2

```

, :
1. , ConTeXt . ; , , , , .
2. .
, „→”, , , .

```

9.2.1

```

, , , , , . ConTeXt , .
:
• , . , ConTeXt , ( ), , , , .
• , , , . ConTeXt ( 4.2.1), , , „Moja fina labela” „Moja fina labela” , .
, , , . („Moja fina labela”) , , 'sec:Labele cilja' .
, :
1. , „reference” .
, ( 7.3), . :
\section[]{}
ConTeXt,
\startsection
[title=, reference=, ... ]

```

(, , .) .

• , , (,), , , , .

• , , ConTeXt . :



```
\chapter[labela1, labela2, labela3] { }
, , ConTeXt .
```

2. \pagereference, \reference, \textreference :

```
\pagereference[]
\reference[]{}
\textreference[]{}

```

- \pagereference .
- \reference \textreference .
- \reference \textreference, (), .
- :

- () . , .
- (, , , .) (, , .)
- , , , \placetable, .
- \pagereference .
- \reference \textreference .
- \reference \textreference. (); , , \reference \textreference .



9.2.2

• , , . , , \goto 9.4.2.

- , ConTeXt ; , :
- \at .
- (, , , .) , \in .
- , (, \placefigure, .) \about .
- , \at \in \about, :

```
\at{}[]
\in{}[]
\about{}[]

```

- .
- . , , () .
- \in , \at .

| | | |
|--|--|--------------------------|
| <pre>\in{}[sec:target labels], \at{}[sec:target labels],</pre> | | <pre>9.2.1, 100, .</pre> |
|--|--|--------------------------|

ConTeXt (9.3), \in \at . , :

| | | |
|--|--|--------------------------|
| <pre>\in{}[sec:target labels], \at{}[sec:target labels],</pre> | | <pre>9.2.1, 100, .</pre> |
|--|--|--------------------------|

, , .

ConTeXt \at, \in \about, , , , „??”.

ConTeXt :

1. .

2. , (4.5.1 4.6).

(), PDF „??” .

• `\ref`

`\at, \in \about` . `\ref`, :

`\ref[][]`

:

- `text:` .
- `title:` .
- `number:` . , .
- `page:` .
- `realpage:` .
- `default:` ConTeXt . `number`.

, `\ref \at, \in \about`, , . , 1500 () , . , `\ref` , `\about`, , .

`\ref` (.), .

•

ConTeXt . „ ” . : . , ConTeXt `\somewhere` :

`\somewhere{ }{ }[]`.

, :

`\type{\somewhere}` .

`\somewhere{}{}[sec:references]`
`\somewhere{}{}[sec:interactivity]`.

•

`\atpage` :

`\atpage[]`

, , , `\atpage` , , .

`\atpage` „precedingpage” , „hereafter” .

, : ConTeXt „ ”, „ ” . ConTeXt , `\atpage`, „¹ () (10.5.3)

, `\atpage` :

`\setuplabeltext[sr][precedingpage=]`
`\setuplabeltext[sr][hereafter=]`

„ConTeXt Standalone” (). ConTeXt `\setuplabeltext`, `\atpage`:

- „precedingpage” „followingpage”.
- „hencefore” „hereafter”.

`\atpage` , , , „precedingpage”, „hereafter”, .

9.2.3

• , „sec:” , „fig:” , „tbl:” , .

, ConTeXt :

- ConTeXt .
- , , ConTeXt.

¹ У српском преводу је за овај појам изабран израз *ознаке*, тако да не долази до овог проблема – прим. прев.

, , , , , ConTeXt , .

9.3

; . , , ; : . , , ; , , .
 ConTeXt (ConTeXt PDF), .
 , ConTeXt PDF, PDF , PDF , .

9.3.1

ConTeXt , . . :

`\setupinteraction[state=start]`

, . „state=start” , „state=stop” , .
 . ConTeXt , . ConTeXt , .

:

- ConTeXt .:

— , , .
 — , .
 — , , .
 — .

- , , , , , , . .:

9.3.2

, `\setupinteraction` ; , . :

- color: .
- contrastcolor: .
- style: .
- title, subtitle, author, date, keyword: PDF ConTeXt.
- click: .

9.4

URL , . :

9.4.1 ,

URL , ConTeXt , , URL, , URL . , URL , URL .
 ConTeXt URL . URL , . URL . :

`\useURL`

URL , , . URL .

:

1. `\useURL[name][URL]`
2. `\useURL[] [URL] [] []`

- , URL , , URL, .

- . URL , , . : ConTeXt: <http://www.pragma-ade.com/general/manuals/what-is-context.pdf>. , :

```

\useURL [WhatIsCTX]
[http://www.pragma-ade.com/general/manuals/what-is-context.pdf]
[]
[ \ConTeXt?]

:

\useURL [WhatIsCTX]
[http://www.pragma-ade.com/general/manuals/]
[what-is-context.pdf]
[ \ConTeXt?]

„WhatIsCTX”, , ; URL , „WhatIsCTX”.

URL \useURL, \url[ ] URL . URL , .

\url URL \setupinteraction, . \setupurl ( style) ( colour).

\hyphenatedurl

URL , , ConTeXt URL . :

\hyphenatedurl{URL}

\hyphenatedurl, URL . URL . :

\sethyphenatedurlnormal{}
\sethyphenatedurlbefore{}
\sethyphenatedurlafter{}

, .

\hyphenatedurl URL . \useURL . :

\useURL [WhatIsCTX]
[http://www.pragma-ade.com/general/manuals/what-is-context.pdf]
[]
[\hyphenatedurl{http://www.pragma-ade.com/general/manuals/what-is-context.pdf}]

\hyphenatedurl :

• % \letterpercent
• # \letterhash
• \ \letterescape \letterbackslash.

\hyphenatedurl , \hyphenatedurlseparator . , URL . :

\def\hyphenatedurlseparator{\curvearrowright}

:

https://support.microsoft.com/?scid=http://support.microsoft.com:80/support/kb/articles/Q208/4/27.ASP&NoWebContent=1.

```

9.4.2

```

\useURL URL \from, , . \useURL . :

\from[]

URL \useURL.

, , \goto . :

\goto{ }[]

, , :

• . \goto , , \in \at . , .
• URL . URL , :

\goto{ }[\url(URL)]

URL , URL \useURL, URL , URL, ConTeXt ConTeXt. URL ConTeXt .

```


9.5 PDF

PDF PDF , .

ConTeXt PDF , `\placebookmarks`, :

`\placebookmarks[]`

.

:

- `\placebookmarks` . (`\starttext \stoptext`, `\startproduct \stopproduct`), : . , , .
 - (`\definehead`) . .
 - , .
 - , „all”, , ; , , , .
- PDF ; , . , PDF . Acrobat, , .

III

: **10.1** ; 10.1.1 ; 10.1.2 ; 10.1.3 ; 10.1.4 ; 10.1.5 ; 10.1.6 ; **10.2** ; 10.2.1 , ; 10.2.2 ; 10.2.3 ; **10.3** ; 10.3.1 ; 10.3.2 ; 10.3.3 ; **10.4** ; **10.5** ; 10.5.1 ; 10.5.2 ; 10.5.3 ; 10.5.4 ; ;
`\translate;` `\quote` `\quotation;`

: , , , .

„” ConT_EXt , .

10.1

UTF-8 (4.1) , . , , UTF-8 . , () , ; , , ConT_EXt .

10.1.1

(), , . () , () (); , , „navegação”.
 T_EX ; (). , , (, ...) .

| | | |
|------|-------------------|------------------------------------|
| ú | <code>\'u</code> | <code>\uacute</code> |
| ù | <code>\`u</code> | <code>\ugrave</code> |
| û | <code>\^u</code> | <code>\ucircumflex</code> |
| () ü | <code>\"u</code> | <code>\udiaeresis, \uumlaut</code> |
| ũ | <code>\~u</code> | <code>\utilde</code> |
| ū | <code>\=u</code> | <code>\umacron</code> |
| ů | <code>\u u</code> | <code>\ubreve</code> |

10.1

10.1 . 'u' , (¹), .

- (3.2), , : 'ã', `\=a \=_a`.² (`\u`), , .
- , , `\emacron` 'e' (ē), `\Emacron` 'E' (Ē), `\Amacron` 'A' (Ā).

10.1 , 10.2.

| | | |
|-------------|------|---|
| O | ø, Ø | <code>\o</code> , <code>\O</code> |
| A | å, Å | <code>\aa</code> , <code>\AA</code> , <code>{\r a}</code> , <code>{\r A}</code> <code>\aring</code> , <code>\Aring</code> |
| L | l, Ł | <code>\l</code> , <code>\L</code> |
| | ß | <code>\ss</code> , <code>\SS</code> |
| 'i' and 'j' | ı, Ĳ | <code>\i</code> , <code>\j</code> |
| | ŭ, Ŭ | <code>\H u</code> , <code>\H U</code> |
| | ç, Ç | <code>\c c</code> , <code>\c C</code> <code>\ccedilla</code> , <code>\Ccedilla</code> |

10.2

, . , (ß), , , ().

A „`{\r A}`” `\Aring` .

'o', 'k', 'l', 'n', 'r', 's' 't', . `\kcedilla`, `\lcedilla`, `\ncedilla` ...

¹ Од свих команди које се налазе у [табели 10.1](#), тилда не функционише са словом 'e' и није ми јасно зашто.

² Упамтите да када је у овом документу важно да се виде, размаке представљамо са '`_`'.

10.1.2

„” , , ’&’, () „et”, (ß), , , ’s’ ’z’. ; . , ’&’
Pagella ’ß’ Bookman.

, (6,) (, 30 pt) .

, , „oe” „ae” ’œ’ ’æ’ . ConT_EXt 10.3

| | | |
|------|-------------------------------------|---|
| æ, Æ | <code>\ae</code> , <code>\AE</code> | <code>\aeligature</code> , <code>\AEligature</code> |
| œ, Æ | <code>\oe</code> , <code>\OE</code> | <code>\oeligature</code> , <code>\OEligature</code> |

10.3

() ’D’: ’D’ ’E’. , ConT_EXt ¹ , 10.1.5.

, , „ ”, ConT_EXt , () `\definefontfeature` () .

10.1.3

. ConT_EXt , . , , , . , `\mu` (μ), `\Mu` (M). 10.4 , .

| (/) | (/) |
|---------|--|
| α, A | <code>\alpha</code> , <code>\Alpha</code> |
| β, B | <code>\beta</code> , <code>\Beta</code> |
| γ, Γ | <code>\gamma</code> , <code>\Gamma</code> |
| δ, Δ | <code>\delta</code> , <code>\Delta</code> |
| ε, ε, E | <code>\epsilon</code> , <code>\varepsilon</code> , <code>\Epsilon</code> |
| ζ, Z | <code>\zeta</code> , <code>\Zeta</code> |
| η, H | <code>\eta</code> , <code>\Eta</code> |
| θ, ϑ, Θ | <code>\theta</code> , <code>\vartheta</code> , <code>\Theta</code> |
| ι, I | <code>\iota</code> , <code>\Iota</code> |
| κ, κ, K | <code>\kappa</code> , <code>\varkappa</code> , <code>\Kappa</code> |
| λ, Λ | <code>\lambda</code> , <code>\Lambda</code> |
| μ, M | <code>\mu</code> , <code>\Mu</code> |
| ν, N | <code>\nu</code> , <code>\Nu</code> |
| ξ, Ξ | <code>\xi</code> , <code>\Xi</code> |
| ο, O | <code>\omicron</code> , <code>\Omicron</code> |
| π, ϖ, Π | <code>\pi</code> , <code>\varpi</code> , <code>\Pi</code> |
| ρ, ϱ, P | <code>\rho</code> , <code>\varrho</code> , <code>\Rho</code> |
| σ, ς, Σ | <code>\sigma</code> , <code>\varsigma</code> , <code>\Sigma</code> |
| τ, T | <code>\tau</code> , <code>\Tau</code> |
| υ, Υ | <code>\upsilon</code> , <code>\Upsilon</code> |
| φ, ϕ, Φ | <code>\phi</code> , <code>\varphi</code> , <code>\Phi</code> |
| χ, X | <code>\chi</code> , <code>\Chi</code> |
| ψ, Ψ | <code>\psi</code> , <code>\Psi</code> |
| ω, Ω | <code>\omega</code> , <code>\Omega</code> |

10.4

(, , , ,) .

10.1.4

, T_EX (ConT_EXt) . . .

10.1.5

, . UTF-8 () . ConT_EXt . , .

, , / . , . `\definecharacter` :

`\definecharacter`

- . , .
- `\.` :

¹ Док у L^AT_EX можемо да употребимо команду `\DH` коју имплементира пакет „fontenc”.

```

— ( ).
— . , , \showfont[ ].
— .
, (10.1.2). : 'D'. , , \decontract \decontract. :
\definecharacter decontract D
, , , , 'D' .
ConTeXt \definecharacter. . :
\buildmathaccent
\buildtextaccent
\buildtextbottomcomma
\buildtextbottomdot
\buildtextcedilla
\buildtextgrave
\buildtextmacron
\buildtextogonek
: , ConTeXt (c, k, l, n, r, s t), . 'b' \buildtextcedilla :
\definecharacter bcedilla {\buildtextcedilla b}
\buildtextcedilla 'b' : 'b'. „” . , .
\buildmathaccent \buildtextogonek . .
\buildtextaccent , . „buildtextaccent”, ; , . .
'D' ,
\definecharacter decontract {\buildtextaccent D E}
'D' („D”) . , .
\definecharacter unusual {\buildtextaccent \_ "}
'_' \unusual.
build — . , 'z':
• \buildtextbottomcomma ('z').
• \buildtextbottomdot ('z').
• \buildtextcedilla ('z').
• \buildtextgrave ('z').
• \buildtextmacron ('z').
, \buildtextgrave \buildtextaccent; , . , :

```

$$\grave{z} - \grave{z}$$

10.1.6

„ConTeXt Standalone” ConTeXt . „cc”, „cow”, „fontawesome”, „jmn”, „mvs” „nav”. :

- **cc** „cc”.
- **cow** „cownormal” „cowcontour”.
- **fontawesome** „fontawesome”.
- **jmn** „navigation 1”, „navigation 2”, „navigation 3” „navigation 4”.

- **mvs** „astronomic”, „zodiac”, „europe”, „martinvogel 1”, „martinvogel 2” „martinvogel 3”.
- **nav** „navigation 1”, „navigation 2” „navigation 3”.

was “wasy general”, “wasy music”, “wasy astronomy”, “wasy astrology”, “wasy geometry”, “wasy physics” “wasy apl”.

```

, :

\usesymbols[]
\showsymbolset[]

: „mvs/zodiac”, :

\usesymbols[mvs]
\showsymbolset[zodiac]

:

Aquarius   ♒   ☾
Aries      ♈   ☿
Cancer     ♋   ♋
Capricorn  ♏   ♏
Gemini     ♊   ♊
Leo        ♌   ♌
Libra      ♎   ♎
Pisces     ♉   ♉
Sagittarius ♐   ♐
Scorpio    ♏   ♏
Taurus     ♉   ♉
Virgo      ♍   ♍

. \symbol . :

\symbol[] []

\usesymbols. , ( mvs/zodiac)

\usesymbols[mvs]
\symbol[zodiac][Aquarius]

„♒”, „”, . \definecharacter .

\definecharacter Aries {\symbol[zodiac][Aries]}

\Aries „♈”.

, , . :

\usesymbols[mvs]
\definesymbol[1][{\symbol[martinvogel 2][PointingHand]}]
\definesymbol[2][{\symbol[martinvogel 2][CheckedBox]}]
\startitemize[packed]
\item \item
\startitemize[packed]
\item \item
\stopitemize
\item
\stopitemize

☑
☑
☑
```

10.2

, , , . 6. , , (). , (.) 12.5.

10.2.1 ,

. ConT_EXt, , , :

- `\word{}`: .
- `\Word{}`: .
- `\Words{}`: ; .
- `\WORD{}` `\WORDS{}`: .

`\cap` `\Cap` : , , , 'x' (6.4.2) , , , (6.5.2) :

1. `\cap` `\Cap` , .
2. , , , , `\cap` `\Cap` .

`\cap` `\Cap` , `\Cap` , . 'caps' , .

,

, `\Cap{}` | , ...
`\cap{}` ...

„UN” () („UN”). `\cap{uN}` `\cap` : , ; `\Cap` .

, , , „” :

`\cap{` | `{\bf }` `\cap{}` | .

`\nocap` `\cap` `\cap` . :

`\cap{When I was One I had just begun,` | WHEN I WAS ONE I HAD JUST BEGUN,
`when I was Two I was \nocap{nearly}` | WAS nearly NEW (A.A. MILNE).
`new (A.A. Milne)}.`

`\cap` `\setupcapitals,` , . `\definecapitals.`

:

`\definecapitals[] []`
`\setupcapitals[] []`

„” `\setupcapitals` . , `\cap.` , `\definecapitals` .

, : „title”, „sc” „style”, „yes” „no”. „title” (), „sc” („yes”), („no”). ,
 . „style” `\cap.`

10.2.2

(3.1) „_” „^” . , `ConTeXt` :

- `\high{}`: .
- `\low{}`: .
- `\lohi{}`: , : , , :

`\lohi{}` |

10.2.3

`verbatim` (`verbum = + atim`), „” „”, `ConTeXt` , . `ConTeXt` `\type.` ,
`typing` . , `ConTeXt` , , , `\type`, `\starttyping` , , .

`\type` : (`ConTeXt`), () .

`ConTeXt \type` , ; , .

:

```
\type i i
\type l l
\type z z
\type ( (
```

'l', 'l', 'z'; , `ConTeXt` `\type` . , , '(' ', ' ' ', `\type` , .

`\type` :

- '{, '}'.

```

    ▪ '<<', '>>'.
    : \type „", \type , ConTEXt; \type, : ConTEXt . : \type * : '}'*.
\type \starttyping \setuptype \setuptyping. \definetype \definetyping. , „setup-en.pdf”
( tex/texmf-context/doc/context/documents/general/qrcs).

\type :
• \typ: \type, .
• \tex: TEX ConTEXt: . , \type . , \tex ConTEXt.

```

10.3

10.3.1

ConT_EXt (T_EX) :

- , , , , , „ ” (”) „ ” (”). , , , .
- , .
- , ConT_EXt , 11.3. , ConT_EXt (), .
- , , ConT_EXt , .

(), . \setupspace[*broad*] \setupspace[*packed*]. (,), 10.5.2.

10.3.2

, . ConT_EXt :¹ \stretched, :

```

\stretched[]{}
:
• factor: . . 0.05 .
• width: , .
  , width factor 0.25, width . \stretched , . 0.25 width, ( 0).
• style: .
• color: .
  , :
\stretched[width=4cm]{\tt test text}
\stretched[width=6cm]{\tt test text}
\stretched[width=8cm]{\tt test text}
\stretched[width=9cm]{\tt test text}
. 'x' 't' „text”, 'e' 'b' „test”, .
, . , \ , . :
\stretched{\tt \}
\setupstretched.
\definestretched , , ( 3.6) \setupstretched \setupcharacterkerning \definecharacterkerning. , \stretched,

```



10.3.3

, , ConT_EXt , 4.2.1. , :

¹ За филозофију система ConT_EXt је врло типично да се обезбеди команда која ради нешто што сама ConT_EXt документација саветује да се не ради. Мада се иде за типографском перфекцијом, циљ је и да се аутору омогући апсолутна контрола над изгледом свог документа: на аутору је одговорност да ли је то боље или лошије.

- `\,` (`.`), `,` (`. 1.000.000`), `:` `„1\,473\,451”` „1 473 451”.
- `\space` „`_`” (`,` , „`_`”, `)` .
- `\enskip`, `\quad` `\qquad` , `1` `2` . `'m'` , `12` , `\enskip` `6` , `\quad` `12` , `\qquad` `24` .
`,` `\hskip` `\hfill` :

`\hskip` . :

```
\hskip 1cm 1 \\  
\hskip 2cm 2 \\  
\hskip 2.5cm 2,5 \\  

, . :
```

```
\hskip -1cm
```

```
\hfill, , , , :
```

```
\hfill \\  
\hfill
```

10.4

„” , . : „” („ + ”) . , (, !). , „—” „—”.

`ConTeXt` . , , .

„||” `ConTeXt` . :

- , „||”.
- / , „|/|”.

`ConTeXt` , . `ConTeXt` `\setuphyphenmark` , „--”) . „||”, `ConTeXt` „—”.

`\setuphyphenmark` (). „--”, “---”, “-”, “(”, “)”, “=”, “/”. , „=” e („---”).

„||” , . , „()”, , “/”. , „(|)|” „|/|”. , „|=|” „|---|” (—).

10.5

. `ConTeXt` , . :

- .
- .
- .

10.5.1

`ConTeXt` . :

- `\mainlanguage` .
- `\language` .

(). `ô` ISO 639-1, , , .

10.5 `ConTeXt`, ISO , , .¹

¹ Табела 10.5 приказује кратак преглед листе која се добија следећим командама:

```
\usemodule[languages-system]  
\loadinstalledlanguages  
\showinstalledlanguages
```

Ако овај документ читате доста касније од времена када је написан (2020), могуће је да `ConTeXt` садржи и додатне језике, тако да би била добра идеја да покренете ове команде и видите ажурирану листу језика.

Током 2020. је додата подршка за српски језик — прим. прев.

| | ISO 639-1 | ISO 639-2 |
|-----------------|---------------------------|--|
| Afrikaans | af, afrikaans | |
| Arabic | ar, arabic | ar-ae, ar-bh, ar-dz, ar-eg, ar-in, ar-ir, ar-jo, ar-kw, ar-lb, ar-ly, ar-ma, ar-om, ar-qa, ar-sa, ar-sd, ar-sy, ar-tn, ar-ye |
| Catalan | ca, catalan | |
| Czech | cs, cz, czech | |
| Croatian | hr, croatian | |
| Danish | da, danish | |
| Dutch | nl, nld, dutch | |
| English | en, eng, english | en-gb, uk, ukenglish, en-us, usenglish |
| Estonian | et, estonian | |
| Finnish | fi, finnish | |
| French | fr, fra, french | |
| German | de, deu, german | de-at, de-ch, de-de |
| Greek | gr, greek | |
| Greek (ancient) | agr, ancientgreek | |
| Hebrew | he, hebrew | |
| Hungarian | hu, hungarian | |
| Italian | it, italian | |
| Japanese | ja, japanese | |
| Korean | kr, korean | |
| Latin | la, latin | |
| Lithuanian | lt, lithuanian | |
| Malayalam | ml, malayalam | |
| Norwegian | nb, bokmal, no, norwegian | nn, nynorsk |
| Persian | pe, fa, persian | |
| Polish | pl, polish | |
| Portuguese | pt, portuguese | pt-br |
| Romanian | ro, romanian | |
| Russian | ru, russian | |
| Slovak | sk, slovak | |
| Slovenian | sl, slovene, slovenian | |
| Spanish | es, sp, spanish | es-es, es-la |
| Swedish | sv, swedish | |
| Thai | th, thai | |
| Turkish | tr, turkish | tk, turkmen |
| Ukrainian | ua, ukrainian | |
| Vietnamese | vi, vietnamese | |

10.5 ConTeXt

ConTeXt uses the ISO 639-1 and ISO 639-2 codes to set the language. The following code sets the language to Spanish:

```
\mainlanguage[es]
\mainlanguage[spanish]
\mainlanguage[sp]
```

ConTeXt also allows you to set the language for specific parts of the document. For example, you can set the language for a specific section:

```
\language[de]
\fr, \de, .) ( 10.5.3).
```

ConTeXt also allows you to set the language for specific parts of the document. For example, you can set the language for a specific section:

```
\setuplanguage[en][patterns={en, agr}]
```

10.5.2

ConTeXt uses the `\setuplanguage` command to set the language. The following code sets the language to Spanish:

```
\setuplanguage[es]
```

ConTeXt also allows you to set the language for specific parts of the document. For example, you can set the language for a specific section:

- `date`: sets the date. (115).
- `lefthyphenmin, righthyphenmin`: sets the number of hyphens. `\setuplanguage[en][lefthyphenmin=4]` (4).
- `spacing`: sets the spacing. (broad), (packed), (spacing=packed), (broad).
- `leftquote, rightquote`: sets the quotes. (), (quote) (116).
- `leftquotation, rightquotation`: sets the quotations. (), (quotation) (116).

10.5.3

ConTeXt uses the `\placetable` and `\placefigure` commands to place tables and figures. The following code places a table:

```
\mainlanguage ( \language)
```



```
\setuplabeltext[] [=]
ConTExT , ConTExT . ,
\setuplabeltext[es][figure=Imagen~]
, \placefigure „Figure x”, „Imagen x”. , „~”; „[figure=Imagen{ }]”
ConTExT .
\setuplabeltext? ConTExT . 2013. ( ) „chapter”, „table”, „figure”, „appendix”... „ ”.
; , ConTExT, „lang-txt.lua”, tex/texmf-context/tex/context/base/mkiv ; ConTExT , .
, \labeltext. , ConTExT \placetable, : „ \labeltext{table} .” \mainlanguage
: „ .”
\setuplabeltext ; „chapter” „section”. ConTExT , , \setuplabeltext[chapter=-] „”.
, ConTExT , \setuplabeltext .
```

10.5.4

```
.
ConTExT . :
• \currentdate: , „ ”. : „11. 2020”. ( , 9/11), (weekday), (day,
month, year)
, „dd” „day” , „mm” ( ) „month” , „MONTH” . , „yy” , „year” „y” . , .
\currentdate[weekday, dd, month]
9. 2020 „ 9 ”.
• \date: , , \currentdate, . , : („d”), („m”) („y”) , ( ) . , e,
6. 1957. ,
\date[d=6, m=7, y=1957][weekday]
.
• \month .
```

.\translate

```
translate , , translate , ( buffer , 12.6):
\startbuffer
\starttabulate[|*{4}{lw(.25\textwidth)}|]
\NC \translate[es=Su carta de fecha, sr= ]
\NC \translate[es=Su referencia, sr= ]
\NC \translate[es=Nuestra referencia, sr= ]
\NC \translate[es=Fecha, sr=] \NC\NR
\stoptabulate
\stopbuffer
, , , . :
\language[es]
\getbuffer

Su carta de fecha Su referencia Nuestra referencia Fecha

\language[sr]
\getbuffer
```

`.\quote \quotation`

`()`, `.`, `ConTeXt \quote \quotation` ; `\quote` , `\quotation` .
`(10.5.2)`; , `-`) , , , :
`\setuplanguage[es][leftquotation=«, rightquotation=»]`.
, ; , `\quote \quotation` `ConTeXt delimitedtext`, `\definedelimitedtext` . :
`\definedelimitedtext`
`[CommasLevelA]`
`[left=«, right=»]`
`\definedelimitedtext`
`[CommasLevelB]`
`[left=", right="]`
`\definedelimitedtext`
`[CommasLevelC]`
`[left=`, right=']`
. , , .
, , (, !).

```

: 11.1 ; 11.1.1 ; 11.1.2 ; 11.2 ; 11.2.1 \setupwhitespace; 11.2.2 ; 11.2.3 ; 11.2.4 \setupblank
\defineblank; 11.2.5 ; 11.3 ConTeXt ; 11.3.1 '~'; 11.3.2 ; 11.3.3 ; 11.3.4 ; 11.4 ; 11.5 ;
11.5.1 ; 11.5.2 ; 11.6 ; 11.6.1 ; 11.6.2 ;

5, , , 6, , , .

```

11.1

ConTeXt. :

1. .
2. `\par \endgraf.`

. (3.7.1) (13.3).

, . : , , , .

. ; , , .

ConTeXt .

11.1.1

. , , \setupindenting :

- always: , .
- yes: . , , .
- no, not, never, none: .

, . (1.5 cm) „small”, „medium” „big” , .

(), ., (. , . ; ()). 12, 12 12 . ConTeXt : \quad , \qquad , . 11 22, 12, 24 .

, , \noindent.

, \setupindenting[yes, big], , , .

11.1.2

, () . , , .

ConTeXt . „narrower” :

`\startnarrower[] ... \stopnarrower`

;

- left: .
- *left: , (2*left).
- right: .
- *right: , (2*right).

```

• middle: . . .
• *middle: , . . .
, „narrower”. \setupnarrower :
• left: .
• right: .
• middle: .
• before: .
• after: .

narrower , \defininenarrower[] []
, \setupnarrower.

```

11.2

11.2.1 \setupwhitespace

4.2.2, ConTeXt : . . . , \setupwhitespace :

```

• none: .
• small, medium, big: , . . .
• line, halflineline, quarterline: , .
• : . . , \setupwhitespace[5pt].
\setupwhitespace. small, medium, big, line, halflineline quarterline. :

```

```

• ( 3.8.2) , , ConTeXt . . .
• small, medium, big, . , , , . , , ( ) .
, : \nowhitespace, \whitespace . , , ConTeXt ; , .
\nowhitespace . \whitespace, ? , , \whitespace . , .

```



11.2.2

```

, \setupwhitespace, , , ConTeXt , . „packed”,
\startpacked[] ... \stoppacked
. , .

```

11.2.3

```

, \blank. , \blank \setupwhitespace. , : small, medium big. , ,
\blank[3*medium] . . , \blank[2*big, medium] insert two large and a medium break.

\blank , ; \blank , ( ). \blank . , „force” . , , ( ),
\chapter :

\setuphead
[chapter]
[
page=yes,
before={\blank[4cm, force]},
after={\blank[3*medium]}
]

. „force”.

```

11.2.4 \setupblank \defineblank

```

\blank , \blank[big]., \setupblank, \setupblank[0.5cm] , \setupblank[medium]. , \setupblank
.

```

```

\setupwhitespace, \blank, , , „fixed”, „flexible”. , , \setupblank[fixed,
line], , \setupblank[flexible, default].

\defineblank. :

\defineblank[] []

, \blank[].
```

11.2.5

```

TEX \vskip. , TEX, ConTEXt ConTEXt. \godown :
\godown[]

. , \godown[5cm] 5 ; 5 , \godown . , \godown , , „\□\godown[3cm]”1 , .
, \blank . , \blank[3cm] \godown[3cm] . , . , \blank , , . , \godown .

TEX ConTEXt, \vfill. . , , \vfill . \vfill , , .

, , :

\vfill
\page[yes]

, \vfill . . :

\page[yes]
\ \vfill

\vfill
\page[yes]
```

11.3 ConT_EXt

15 , ConT_EXt , .

ConT_EXt. , , . (."). , ConT_EXt , , ; ConT_EXt.

ConT_EXt *badness* () . , , ConT_EXt (). , , .

11.3.1 ‘~’

‘~’, T_EX , .

:

- U~S.
- , ~ .~45.
- , ~II, 45~.
- , . , 73~, \$~53; , 35'.
- , ~.
- , 5~357~891. , ConT_EXt \,, 5\,357\,891.
- . :

‘~’

‘~’

, (T_EX) :

- .
- , , , , ~12.

¹ Присетите се да у овом документу карактер ‘□’ користимо за представљање размака онда када је важно да га учимо.

- `~E.`, `A.~`.
- `~d`, `~w`.
- `{1,~2, \dots,~n}`.
- `0 ~1`.
- `~n`.
- `(1)~`, `(2)~`, `(3)~`.

? , . , , . – – (), . (): , , .

11.3.2

, `ConTeXt` (); , , `\mainlanguage`.
`ConTeXt` (, `ConTeXt`); . , `ConTeXt` „unionised”? „the unionised workforce” (), „an unionised particle” (.). `ConTeXt` „manslaughter” () , . man-slaughter () mans-laughter ().

, , , `\-` , „unionised” , „union\~ised”; „manslaughter”, „man\~slaughter”.
 , `\hyphenation:` , , . :

`\hyphenation{union-ised, man-slaughter}`

, `\hbox` , `\unhyphenated` . `\hyphenation` , `\hbox` `\unhyphenated` ,
`\hyphenation` ; `\hbox` `\unhyphenated` .

, `TeX` `\pretolerance` `\tolerance`. 100, , 10000, `ConTeXt` , *de facto* . `\pretolerance` -1, ,
`ConTeXt` .

`\pretolerance` . :

`\pretolerance=10000`

„lessshyphenation” „morehyphenation” `\setupalign.` , 11.6.1.

11.3.3

, `ConTeXt` , , . () , , `\break` , , .
 , `ConTeXt` . `\setuptolerance` `ConTeXt` „” () „” . 11.6.2.

() „verystRICT”. , , : „strict”, „tolerant”, „verytolerant” „stretch”. ,

`\setuptolerance[horizontal, verytolerant]`

, .

11.3.4

, `\break, \crlf` `\` . , `\break`, . `\break` () ().

`\emph{}`
`\break` ,

,

, `\` `\crlf` , , :

`\emph{}`
`\` ,

,

, `\` `\crlf`; :

- `\` , .

- `\crlf` .

. , , . :

- `ConTeXt` , . (, , [10.2.3]), `\break` .
- , , , , ; , , , . `\\ \crlf.` .
- , , . `writing lines` 11.5.1.

11.4

- `ConTeXt` `\setupbodyfont \switchtobodyfont.`
- `\setupinterlinespace` :
 - , (, 15pt), (, 1.2). „ ” `ConTeXt` .
 - , „small”, „medium” „big” , , , `ConTeXt`.
- `\setupinterlinespace [...]=...]` , `ConTeXt` . ; : , . `\setupinterlinespace` , . (,) : `line, height, depth, minheight, mindepth, distance, top, bottom, stretch shrink.`
- `\setupinterlinespace [].` `\defineinterlinespace.`

```
\defineinterlinespace[] []
\setupinterlinespace[], \setupinterlinespace[reset].
```

11.5

11.5.1

(4.2.2), `ConTeXt` , , , , , . `ConTeXt` „lines” :

```
\startlines[] ... \stoplines
```

- `space:` „on”, , , .
- `before:` .
- `after:` .
- `inbetween:` .
- `indenting:` (11.1.1).
- `align:` (11.6).
- `style:` .
- `color:` .

```
,
\startlines
,
.
.
.
\stoplines
\setuplines, ConTeXt, \definelines . :
```

```
\definelines[] []
```

- , :

`\startlines[] ... \stoplines`

11.5.2

`\startlinenumbering[] ... \stoplinenumbering`

ConTeXt linenumbering

`\definelinenumbering[] []`

Options:

- **continue:** („continue=no”, „continue=yes”).
- **start:** '1', ...
- **step:** (5, 10, 15, ...).
- **linenumbering:** `\setuplinenumbering` `linenumbering` :
- **conversion:** 89 .
- **style:** () (, , ...).
- **color:** .
- **location:** : text, begin, end, default, left, right, inner, outer, inleft, inright, margin, inmargin.
- **distance:** .
- **align:** : inner, outer, flushleft, flushright, left, right, middle auto.
- **command:** .
- **width:** .
- **left, right, margin:** `\definelinenumbering` :

`\definelinenumbering[] []`

`\startlinenumbering[] ... \stoplinenumbering`

11.6

`\setupalign.`

11.6.1

1. `\setupalign` `flushright` `flushleft` `flushright` `flushleft`, `\setupalign` `right` `left` : `left` „”, `right` „”, `\setupalign[left]` , and `\setupalign[right]` .

ConTeXt: „ConTeXt flushleft flushright.”, ConTeXt, ' ' ' ' , ' ' ' ' ,

2. `\setupalign` middle.

3. ConTeXt, `\setupalign` , , `\setupalign[reset]`.

`\setupalign` (right, flushright, left, flushleft, inner, flushinner, outer, flushouter middle) broad,

`\setupalign` , , ; .

¹ Под изразом *тачна* мислим на ширину линије пре него што ConTeXt подеси величину размака између речи како би омогућио поравнање.

```
\setupalign morehyphenation ConTEXt lesshyphenation . \setupalign[horizontal, morehyphenation],
\setupalign[horizontal, lesshyphenation] , .
```

```
\setupalign , , . , :
```

- „alignment”, . :

```
\startalignment[] ... \stopalignment
```

```
\setupalign.
```

- \leftaligned, \midaligned \rightaligned , , ; (,) , \wordright. .
, „right” „left” \setupalign , \leftaligned \rightaligned : left , right .

11.6.2

, : , ? („height”), ; („bottom”) („line”). „bottom”.

```
\setuptolerance ConTEXt ( ), , . ConTEXt . : verystrict, strict,
tolerant verytolerant. \setuptolerance [vertical, strict].
```

. . , ; , .

```
ConTEXt . ConTEXt: \widowpenalty , \clubpenalty . , :
```

```
\widowpenalty=10000
\clubpenalty=10000
```

```
ConTEXt . . 10000, , ; , , 150 , , .
```

```

: 12.1 ; 12.1.1 ConTeXt ; 12.1.2 ; 12.1.3 ; 12.1.4 ; 12.1.5 ; 12.1.6 ; 12.2 ; 12.2.1 \startcolumns
; 12.2.2 ; 12.3 ; 12.3.1 ; ; ; 12.3.2 ; 12.3.3 ; 12.3.4 ; 12.3.5 \items ; 12.3.6 ; 12.4 ; 12.4.1 ;
12.4.2 ; 12.5 ; 12.5.1 ; 12.5.2 ; 12.5.3 ; 12.6 ;

```

12.1

” [Libro de Estilo de la Lengua española (), . 195]. ();

(), ConTeXt

, :

- : . , , :
- : , . , , . ,
- , ,
- : ,

12.1.1 ConTeXt

ConTeXt . , . :

- : , \footnote.
- : \endnote ; , (\placenames).
- : , ID . () \inmargin \marginintext.
- : , \startlinenumbering ... \stoplinenumbering (11.5.2). , , . \linenote
\setuplinenote. , ().
- :
- (5.7).
- () ,
- , () () ConTeXt,

12.1.2

, , (). \defineline (12.1.4) .

:

```

\footnote[]{}
\endnote[]{}

```

-
- , ,

```

, \note . . :

,\footnote[glisa]{
}
\note[glisa]\\
.\\ ,
.\\
, .

\footnote \endnote :

\footnote , , ( , ) . , ConTeXt .

,\footnote , . , , columns . , \footnote \footnotetext \note . , . , \note . , . ,
\note[MyLabel] , \footnotetext[MyLabel]{ }.

\footnotetext \note . :

%
\footnote{ \note[noteB] , .}%
\footnotetext[noteB]
{ \note[noteC] .}
\footnotetext[noteC]{
.}
.

1 Врло популаран карактер из српске дечије песмице

2
2 или ово3, ако вам више одговара.
3 или можда чак и ово4.
4 би могло бити нешто потпуно другачије.

\endnote . , (\placenotes[endnote]) , , ( ).

```

12.1.3

```
\startlocalfootnotes , , \placelocalfootnotes, .
```

12.1.4

```

\definernote . , , .

\definernote :

\definernote [] [] []

• .

• . footnote endnote; , , \placenotes[endnote] \placenotes [] ( ).

• : , , .

ConTeXt ( 3.6) \setupnote . , : \setupnote \setupnotation. , .

, „BlueNote” , :

\definernote [BlueNote] [footnote]
\setupnotation
[BlueNote]
[color=blue, style=bf]

, . BlueNote, . \BlueNote \footnote:

\BlueNote [] {}

```

12.1.5

```
( , \definernote \linenote) : \setupnote \setupnotation5. :
```

⁵ \setupnote има 35 директних опција конфигурације и још 45 додатних наслеђених од \setupframed; \setupnotation има 45 директних конфигурационих опција и још 23 наслеђених од \setupcounter. Пошто ове опције нигде нису документоване, и мада за већину њих из њиховог имена можемо наслутити њихову корисност, морамо проверити да ли је наша интуиција у праву или није; и такође, узимајући у обзир да многе од ових опција прихватају већи број вредности које све треба да се тестирају... Да бих написао ово објашњење, видећете да сам морао извршити поприличан

```

\setupnote[] []
\setupnotation[] []

    (footnote, endnote      ) .

    , ; . , , ,1 , , \setupnotation , , \setupnote .

    . , :

• :

. :

– : start \setupnotation. , .

– , numberconversion \setupnotation. :

* : n, N numbers.
* : I, R, Romannumerals, i, r, romannumerals. , .
* : A, Character, Characters, a, character, characters ( ) ().
* . , , , '3' 'three'. Words , words .
* : , : set 0, set 1, set 2 set 3. 90 .

– : way \setupnotation. bytext, , . bychapter, bysection, bysubsection, ., ,
, byblock ( 7.6) . bypage, .

• :

– : number \setupnotation.

– : alternative \setupnotation: : left, inleft, leftmargin, right, inright, rightmargin,
inmargin, margin, innermargin, outermargin, serried, hanging, top, command.

– : numbercommand \setupnotation.

– : textcommand \setupnote.

numbercommand textcommand . . , (\bf, \it, .), .

– ( ): distance width \setupnotation. ( ) .

– : interaction \setupnote. yes , no .

• .

:

– : location \setupnote.

(location=page), \placenotes[endnote] (location=text). , location=text, \placenotes[footnote],
\placefootnotes.

– : , paragraph \setupnote „yes”.

– : style \setupnotation.

– : bodyfont \setupnote.

```

број тестова; и мада је извршавање теста брзо, обављање великог броја тестова је споро и досадно. Тако да се надам да ће ми читалац опростити ако вам кажем да ћу осим две опште конфигурационе команде за напомене које помињем у главном тексту и на које се фокусирам у објашњењу које следи, изоставити остале четири потенцијалне конфигурационе могућности из објашњења:

- `\setupnotes` и `\setupnotations`: другим речима, исто име али у множини. Вики каже да су верзије команде у једнини и множини синоними, и ја верујем да је тако.
- `\setupfootnotes` и `\setupendnotes`: претпостављамо да су ово специфичне примене за фусноте и напомене на крају, редом. Можда би објашњавање конфигурације напомене на основу ових команди било лакше, међутим, пошто прва опција (`numberconversion`) коју сам покушао са `\setupfootnotes` није успела да проради, мада знам да остале опције ових команди функционишу... био сам сувише лењ да многим тестовима које само морао да урадим како би написао оно што следи додам и тестове потребне да се у објашњење укључе и ове две команде. Али мишљења сам (након неколико случајних тестова које сам извршио) да све што ради у ове две команде, али чије објашњење изостављам, такође ради и у командама за које дајем објашњење.

¹ Постоји страница у [ConTeXt викију](#) коју сам случајно открио (јер није директно посвећена напоменама), која сугерише да је разлика у томе што `\setupnotation` контролише текст напомене који треба да се уметне, а `\setupnote` окружење напомене у којем ће се она поставити (?) Али то није у складу са чињеницом да се, на пример, ширина текста напомене (која има везе са њеним *уметањем*) контролише опцијом `width` команде `\setupnote`, а не `\setupnotation` опцијом са истим именом. Оно што се овде контролише је ширина простора између маркера и текста напомене.

```

        . , ConTeXt
- : margin \setupnotation.
- : width \setupnote.
- : n \setupnote . 'n' .
• : :
- rule, \setupnote (rule) . yes, on, no off. , .
- before, \setupnotation: . , , .
- after, \setupnotation: .

```

12.1.6

```

\notesenabledfalse \notesenabledtrue ConTeXt . , . , , , , , .

```

12.2

```

:
. .
. , , , , .
. .
, . ; ( 13.1) ( 13.3) .
, ConTeXt ., :
• : ( ) ( ) . , , .
• : ConTeXt ( ), 20 40 .
,
• ( 5.3.4).
• , , , .

```

12.2.1 \startcolumns

```

columns :
\startcolumns[] ... \stopcolumns
. , ,
\setupcolumns[Configuration]
. , , :
• :
- n: . , .
- nleft, nright: ( 5.3.4), () ().
- distance: .
- separator: . space () rule, (rule) . , :
* rulecolor: .
* rulethickness: .
- maxwidth: + .
• :

```

- `balance`: `ConTeXt` , , `no` .
- `direction`: . (), `reverse` .
- :
- `tolerance`: , `ConTeXt` (11.3), . (11.3.3), .
- `align`: . : `right`, `flushright`, `left`, `flushleft`, `inner`, `flushinner`, `outer`, `flushouter`, `middle`, `broad` . , 11.6.1.
- `color`: .

12.2.2

```

. ( , ), , , , , , ; , , .
; ConTeXt. \defineparagraphs \setupparagraphs , , , .
, . „" ConTeXt : „paragraphs”.

\defineparagraphs :
\defineparagraphs [] []

, ,

\setupparagraphs [] [] []

, , .

:

\defineparagraphs
[MursijaCinjenice]
[n=3, before={\blank},after={\blank}]

\setupparagraphs
[MursijaCinjenice][1]
[width=.1\textwidth, style=bold]

\setupparagraphs
[MursijaCinjenice][2]
[width=.4\textwidth]

MursijaCinjenice, 10 . , , 50%.

, :

\startMursijaCinjenice
825
\MursijaCinjenice
.
\MursijaCinjenice
825. - II
( ) ,
.
\stopMursijaCinjenice

825 . , 825. - II ( ) , .

, (\startMursijaCinjenice), .

:

• , \defineparagraphs [BrankoKockica], \startBrankoKockica \stopBrankoKockica.
• \BrankoKockica, .

(\setupparagraphs), , , , , , , \setupparagraphs [] [] , \setupparagraphs
[] [] [] .

• n:
• before:
• after:
• width:
• distance:
• align: \setupalign
• inner:
• rule: on off
• rulethickness:
• rulecolor:
• style:
• color:

```


; . . . , . . . , columns itemize , . . .

12.3

, . . . , . . . , . . . ConTeXt , itemize . . . , . . .
 (), . . . ” . . . () , . . . ; ¹ . . . HTML . . . ; . . .
 ConTeXt , . . . ; ,
 4, . . . , 2013. . . , (15). . . , . . . ; ConTeXt . . .
 , . . . , . . . , . . . ;
 ConTeXt \itemize :
 \startitemize[] [] ... \stopitemize
 . . . ConTeXt ; , . . . , . . .

12.3.1

• . . .
 itemize :
 • . . . ○ . . .
 — . . . ○ . . .
 * . . . □ . . .
 ▷ . . . ✓ . . .
 ° . . .
 , . . . \startitemize[4] ▷ . . .
 \definesymbol:
 \definesymbol[] { }
 \definesymbol[1] [\diamond]
 ◇. . .
 \definesymbol[10] [\copyright]
 10 : ©.
 •
 , itemize . . . :

| | | |
|--------------------------|---------------------------|-------------------------------|
| n 1, 2, 3, 4, ... | a a, b, c, d, ... | r i, ii, iii, iv, ... |
| m 1, 2, 3, 4, ... | A A, B, C, D, ... | R I, II, III, IV, ... |
| g α, β, γ, δ, ... | KA A, B, C, D, ... | KR I, II, III, IV, ... |
| G A, B, Γ, Δ, ... | | |

n m : **n** , **m** , , .²
m,²

12.3.2

\startitemize \item, . . . , Mark IV : \startitem... \stopitem. :

| | |
|--------------------------|----|
| \startitemize[a, packed] | a. |
| \startitem \stopitem | b. |
| \startitem \stopitem | c. |
| \startitem \stopitem | |

¹ Опција `noitemize` се увлачење примењено на све линије пасуса осим на прву назива *висеће увлачење*, чиме се лако проналази прва реч или први карактер пасуса.

² Опција `m` укључује такозване *oldstyle* бројеве. То је стилска алтернатива уграђена у неке фонтове (*opim* OpenType особина) која може да се укључи командом `\os.` — прим. прев.

```

\startitemize[a, packed]
\item
\item
\item
\stopitemize

```

a.
b.
c.

```

\item \startitem . :

\head \item .

, . \item \head ConTeXt .

\sym \sym{ } . \sym \item.

\sub ( ). , , '+'. , , .

\mar , ( , ).

, , . \ran ( range - ) \its, items - . ( ), x ( 4, items). :

, :

\startitemize[5, packed][width=8em, distance=2em, items=5]
\ran{ \hss }
\its \ConTeXt, .
\its .
\its .
\its .
\its .
\quotation{ } .
\stopitemize

, :

o o o o o ConTeXt, .
o o o o o .
o o o o o .
o o o o o " , .

```

12.3.3

- „itemize” . ; „itemize” :
- columns: . columns : two, three, four, five, six, seven, eight nine. columns , .
 - intro: .
 - continue: () . continue , .
 - packed: .
 - nowwhite: packed, : , .
 - broad: . broad, example \startitemize[2*broad].
 - serried: .
 - intext: .
 - text: .
 - repeat: . : 1, 2, 3, 4; : 1.1, 1.2, 1.3, . , .
 - margin, inmargin: , (atmargin). margin inmargin .

12.3.4

- ```
\startitemize, , .
```
- before, after: itemize .
  - inbetween: .

- `beforehead, afterhead:` `\head.`
- `left, right:`  `, , :`  
`\startitemize[a][left=(, right=)]`
- `stopper:`  `. .`
- `width, maxwidth:`  `, .`
- `factor:`  `.`
- `distance:`  `.`
- `leftmargin, rightmargin, margin:` `(leftmargin) (rightmargin) .`
- `start:`  `.`
- `symalign, itemalign, align:`  `. \setupalign. symalign ; itemalign , align .`
- `indenting:`  `. 11.1.1`
- `indentnext:`  `. yes, no auto.`
- `items:` `\its, .`
- `style, color; headstyle, headcolor; marstyle, marcolor; symstyle, symcolor:` `\item, \head, \mar \sym.`

### 12.3.5 `\items`

`itemize` `\items :`

`\items[] { 1, 2, ..., n }`

`. :`

`, , :`  
 `, :`

`\items{png, jpg, tiff, bmp}`

`, , :`  

- `png`
- `jpg`
- `tiff`
- `bmp`

`itemize, :`

- `symbol:` `itemize.`
- `n:`  `.`

### 12.3.6

`. , .`

`:`

- `itemize \items .`
- `. : ListAlpha, ListRoman), .`

`\setupitemize \setupitems. \defineitemgroup, \defineitems. itemize, items.`

## 12.4

`, .`

### 12.4.1

`. :`

`\definedescription[] []`

```

,

\definedescription[Concept]

:

\Concept{}
\startConcept {} ... \stopConcept

,

:

\definedescription
[Concept]
[alternative=left, width=3.5cm, headstyle=bold]
\Concept{}

,
\ConTeXt.

:

, \ConTeXt.

\ConTeXt, \setupdescription:
\setupdescription[] []

,

• alternative: left, right, inmargin, inleft, inright, margin, leftmargin, rightmargin,
innermargin, outermargin, serried, hanging , , .

• width: alternative, .

• distance: .

• headstyle, headcolor, headcommand: : (headstyle) (headcolor). headcommand . :
headcommand=\WORD .

• style, color: .

```

### 12.4.2

```

, text. , :

• .

• .

, , , , .

\defineenumeration[] []

, .

:

\defineenumeration
[Exercise]
[alternative=top, before=\blank, after=\blank, between=\blank]

Exercise , , :

\Exercise
\startExercise

exercise () , . , :

\subExercise
\startsubExercise
\stopsubExercise
\subsubExercise
\startsubsubExercise

```

```

\stopsubsubExercise
\subsubsubExercise
\startsubsubsubExercise
\stopsubsubsubExercise

:

• \set: .

• \reset: .

• \next: .

, \setupenumeration :
\setupenumeration[] [].

. , \setupenumeration [subExercise] [] „Exercise”.

\setupenumeration \setupdescription.

```

## 12.5

ConTeXt T<sub>E</sub>X , . : ConTeXt ( T<sub>E</sub>X) . , , ConTeXt . ConTeXt  
 MetaPost, TiKZ (ConTeXt ), .  
 , , .

### 12.5.1

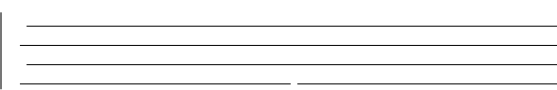
```

\hairline .

\hairline . , \thinrule. . \hairline, \thinrule \hfill (10.3.3),
(\hfill), .

\thinrule\
\thinrule \
\thinrule \
\thinrule \thinrule

```



```

\thinrules . \thinrules[n=2] , . \thinrules , , , \setupthinrules. (rulethickness),
(color), (background), (interlinespace),.

. setup-en.pdf (3.6). (.), , . : rulethickness. height depth.

\hl \v1. , . . \hl (), \v1 .

\hl[3] 3, \v1[3] . , . , 1.

\fillinline . (\setupfillinlines) (width) .

. :

\fillinline[width=6cm]

```



```

., fillinline .

, (margin), (distance), (rulethickness) (color).

\fillinrules \fillinline, („n”).

\fillinrules[] {} {}

```

### 12.5.2

, , ConTeXt :

- .

- `()`, `()` `()`.

`\textrule`. `( )`. :

`\textrule{ }`

`\textrule` : top, middle bottom. , .

`\textrule \starttextrule` , , . :

`\starttextrule[] { } ... \stoptextrule`

`\textrule \starttextrule \setuptextrule`.

, , , :

`\underbar{}`  
`\underbars{}`  
`\overbar{}`  
`\overbars{}`  
`\overstrike{}`  
`\overstrikes{}`

, ( , , ) . , , , .

, , , , `\underbar` , .

`\underbar` . `\underbar` .

### 12.5.3

, :

- `\framed \inframed` .

- `\startframedtext` .

`\framed \inframed` . `\frame` , , . `\inframed` , . :

`\framed{}`  
`\inframed{}`.

`framed` `inframed` `\setupframed`, `\startframedtext` `\setupframedtext`. . (height, width, depth),  
 (framecorner), `rectangular()` (round), (framecolor), (framethickness), (align), (foregroundcolor),  
 (background backgroundcolor), .

`\startframedtext` : frame=off ( , ). , framedtext , .

`\defineframed \defineframedtext`.

## 12.6

ConTeXt , . :

- `buffer` . `\startbuffer[] ... \stopbuffer` `\getbuffer[]`, .
- `chemical` . TeX, , , ConTeXt , .
- `combination` .
- `formula` .
- `hiding` , . , , `\starthiding`, `\stophiding`.
- `legend` , TeX . , `\startlegend` .
- `linecorrection` ConTeXt , . `\framed`. .
- `mode` . , . `\startnotmode`.
- `opposite` .

- `quotation` `narrower`, . , . — .
  - `standardmakeup` , , , .  
    ( ), :
1. `ConTeXt` . ; .
  2. .
  3. `ConTeXt` ( 3.6) , .

,

```

: 13.1 ?; 13.2 ; 13.2.1 ; 13.2.2 \placefigure; 13.2.3 ; 13.2.4 ; ; ; 13.3 ; 13.3.1 ;
13.3.2 tabulate ; 13.4 , ; 13.4.1 ; 13.4.2 ; 13.4.3 ; 13.4.4 ; 13.5 ;

. , , : . , : , , . : ; , , , .

```

## 13.1 ?

, : , . , , , , .

, , , , , , .

; . ” , . ConTeXt , , ; .

, . . , , .

, , ; , , . , : ” ” , , : , .

( ), ” ” ”, ” 1.3”, ConTeXt ( 9.2). , ( , , , .). , ( 8.2).

ConTeXt . : . , .

## 13.2

( 1.5), ConTeXt MetaPost . ConTeXt<sup>1</sup> TiKZ.<sup>2</sup> ( ). , ConTeXt .

### 13.2.1

```

() \externalfigure
\externalfigure[] []


```

- , , \useexternalfigure \externalfigure .
- . 13.2.4.

pdf, mps, jpg, png, jp2, jbig, jbig2, jb2, svg, eps, gif tif. ConTeXt , (svg, eps, gif tif) , , ConTeXt .

\externalfigure .: QuickTime ( .mov), Flash Video ( .flv) MPEG 4 ( .mp4). PDF PDF . , .

: ConTeXt . , , PDF , , MPS ( MetaPost). , : jpeg, png, jpeg 2000, jbig jbig2. ConTeXt method.

, , , , , .

ConTeXt , . directory \setupexternalfigures, . , location. , „img”, :

```

\setupexternalfigures
[directory=img, location=global]

directory \setupexternalfigures , ; . „directory={img, ~/slike}”.

```

<sup>1</sup> ConTeXt модули проширења обезбеђују додатне могућности, али нису део овог увода.

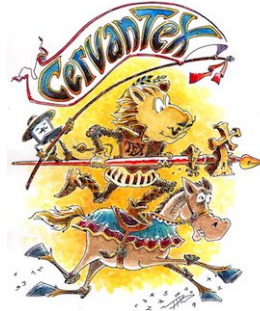
<sup>2</sup> Ово је графички програмски језик намењен за рад са системима базираним на Т<sub>ε</sub>X систему. То је „рекурзивни акроним” немачке реченице „TiKZ ist keinen Zeichenprogramm” што у преводу значи: „TiKZ није програм за цртање”. Рекурзивни акроними су врста програмерске шале. На страну MetaPost (који не знам како да користим), верујем да је TiKZ одличан систем за програмирање графике.



directory '/' ; Microsoft Windows '\.

`\externalfigure` . , CervanTeX . T<sub>E</sub>X :<sup>1</sup>

```
\externalfigure
[http://www.cervantex.es/files/
cervantex/cervanTeXcolor-small.jpg]
```



, LuaT<sub>E</sub>X . . 1 . , ConT<sub>E</sub>Xt .

ConT<sub>E</sub>Xt , , ( ConT<sub>E</sub>Xt ) , . 13.4.3.

### 13.2.2 `\placefigure`

. `\placefigure` . ConT<sub>E</sub>Xt:

- , .
- , .
- , .
- ( ) .
- , .
- , .<sup>2</sup>

:

`\placefigure[] [] {} {}`

:

- . , ( 13.4.1). , here . ConT<sub>E</sub>Xt , , .
- , ( 9.2).
- .
- .

```
\placefigure
[here]
[fig:texnuth]
{\TeX\ {\sc {} }}
{\externalfigure[https://i.ytimg.com/vi/8c5Rrfabr9w/maxresdefault.jpg]}
```

```
name:
https://i.ytimg.com/vi/8c5Rrfabr9w/maxresdefault.jpg
file:
https://i.ytimg.com/vi/8c5Rrfabr9w/maxresdefault.jpg
13.1. TEX
state: unknown
```

, ( , ) , `\externalfigure` . , . , . :

<sup>1</sup> Интернет адресе су веома дугачке, па нема довољно места да се испише у примеру у две колоне. Из тог разлога сам у веб адресу уметнуо прелом реда како би могла да стане у колону. Ако неко жели да копира и налепи пример, он неће функционисати док се не обрише овај прелом линије.

<sup>2</sup> Ово друго је мој закључак, узевши у обзир да се међу опцијама постављања налазе и `force` или `split` које се косе са смислом плутајућих објеката.

- `„”`, `ConTeXt` . `\placelist[figure] ( 8.2),` `\placelistoffigures`  
`\completelistoffigures.`
  - `\placefigure,` ( 9.2).
  - `()`, `ConTeXt` .
- `\placefigure`, `,` `.` `,` `,` `\placefigure,` `,` `;` `ConTeXt` `,` `,` `.` `:`

13.2 `\placefigure`

```

:
\placefigure
[here, force]
[fig:testtext]
{ \backslash placefigure }
{\rotate[rotation=180]{\framed{\tfd }}}

```

### 13.2.3

`figuretext`, `( , )`. `\placefigure`, ( 13.4.1), `ConTeXt` `,` `,` `,`

```

\startfiguretext
□
□
□
□
...

```

```

\stopfiguretext

```

```

\placefigure , ; „left” , , „left, bottom” .

```

### 13.2.4

- `\externalfigure` `.` `:`
- `;` `.` `\setupexternalfigures.`
- `.` `\useexternalfigure` `.`
- `.` `\externalfigure` `.`
- `:`
- `:`
- `,` `width height;` `,` `.`
- `,` `.` `:`
- `width=.4\textwidth`
- `40%` `.`
- `: xscale ; yscale , scale . 1000.: scale=1000 , scale=500 , scale=2000 .`
- `,` `maxwidth maxheight . maxwidth=.2\textwidth 20%` `.`
- `orientation.` `.` `.`
- `90 (90, 180 or 270), , \rotate. , 45 orientation=45, \rotate .`
- `frame=on, framecolor), frameoffset), rulethickness) framecorner) round) .`
- `,` `\setupexternalfigures` `,` `( directory), (location=global),`  
`(location=local), (interaction), .`

•

`ConTeXt` . `\externalfigure` . :

- : `\mirror`.
- : `\clip` (width), (height), (hoffset) (voffset) . :

```
\clip
[width=2cm, height=1cm, hoffset=3mm, voffset=5mm]
{\externalfigure[logo.pdf]}
```

- . `\rotate` . `\externalfigure` , , orientation .

, . , ( ), :

```
\mirror{ }\\
\rotate[rotation=20] { }
```

## 13.3

### 13.3.1

, , . , : ( ) , ( ) . , 13.3.

, , , ( ) `ConTeXt`. (

) , `ConTeXt`. , .

`ConTeXt` ; `tabulate` , `TeX` ;

, `HTML` , , , ; `XML`

. , . „`ConTeXt Mark IV an excursion`”, „`ConTeXt Standalone`” .

: , `\placetable`

. :

- `ConTeXt` ( ), .
- , .

`\placetable` `\placefigure` ( 13.2.2):

`\placetable[] [] {} {}`

13.4.1 13.4.2 . , .

„split”, , `ConTeXt` , .

`\setuptables` . , -

`\placelistoftables`

`\completelistoftables` . , 8.2.2.

13.3

### 13.3.2 tabulate

`tabulate` , :

```
\starttabulate[]
... %
...
...
\stoptabulate
```

( ) , ( ) . , : , . , .

. , , 1 2. , .

1. : „|” . , „[|T|rB|]” , „1” „T” ( ), „r” „B” . , „[|1|1|1|1|]”.

2. : , . `tabulate` , .



• : , , , ( „l”, *left*), ( „r”, *right*) ( „c”, *center*).  
 , „w()”. „[|rw(2cm)|c|c|]” , , 2 , .

• : „P” „P” „P()”, . :

```
\starttabulate[|l|r|p|]
\starttabulate[|l|p(4cm)|]
\starttabulate[|r|p(.6\textwidth)|]
\starttabulate[|p|p|p|]
```

, , , , ; , , ,  
 , „P” , .

3. : ( , ), „B” „I” „S” „R” „T” .

4. :

- : „m” „M” .  
 T<sub>E</sub>X, ConT<sub>E</sub>Xt, ( ) ConT<sub>E</sub>Xt , . : ( „m”) .
- : „in”, „jn” „kn” („in”), („jn”) („kn”). „n” ( ). „|j2r|” 1 .
- . b{ } a{ } („b”, *before*) („a”, *after*) .
- „B”, „I”, „S”, „R” „T” : . , , „f\” . „|lf\sc|” .
- , „h\” .

### 13.1

---

```
|l| .
|rB| , .
|cIm| . , .
|j4cb{---}| , (—) 2 .
|l|p(.7\textwidth)| : , . 70% .
```

---

13.1 tabulate

- , . :
- . \HL ( *Horizontal Line*).
  - : , . \VL ( *Vertical Line*). , , \VL.
  - \NR ( *Next Row*). \HL .
  - , , . \NR \HL .  
 , \HL \VL \NC ( *New Column*).  
 , ô . 13.2 ( ) . , .

---

```
\HL
\NC
\NR
\VL (\NC)
\NN (\NC)
\TB
\NB
```

---

13.2

, , ô 13.2.

```
\placetable
[here, force]
[tbl:tablecommands]
{
}
{\starttabulate[|l|p(.6\textwidth)|]
```

```

\HL
\NC {\bf }
\NC {\bf }
\NR
\HL
\NC \tex{HL}
\NC
\NR
\NC \tex{NC}
\NC
\NR
\NC \tex{NR}
\NC
\NR
\NC \tex{VL}
\NC (\tex{NC})
\NR
\NC \tex{NN}
\NC (\tex{NC})
\NR
\NC \tex{TB}
\NC
\NR
\NC \tex{NB}
\NC
\NR
\HL
\stoptabulate}

```

( ) . ; . , , . , „\NC ” . , ( ) \NR , . , \HL . , !

ConT<sub>E</sub>Xt . \tex 10.2.3.

## 13.4 ,

, , \placefigure \placetable. , , ConT<sub>E</sub>Xt \placechemical ( ), \placegraphic ( ) \placeintermezzo ConT<sub>E</sub>Xt , . \placefloat :

\placefloat[] [] [] {} {}

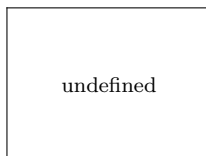
\placefloat \placefigure \placetable \placefloat . : \placefloat[] \place. : \placefloat[figure] \placefigure , \placefloat[table] \placetable.

\placefloat, \placefigure \placetable, .

\placefloat :

- *Name.* . (figure, table, chemical, intermezzo) \definefloat ( 13.5).
- *Options.* ConT<sub>E</sub>Xt . . . .
- *Label.* .
- *Title.* . , 13.4.2.
- *Contents.* . \externalimage; , ; , ; .

. ( ) , . , : \placefloat{}{}



13.4



: ConT<sub>E</sub>Xt , . . . .

### 13.4.1

*Options* \placefigure, \placetable \placefloat . . . , :

- (top, bottom inleft, inright, inmargin, margin, leftmargin, rightmargin, leftedge, rightedge, innermargin, inneredge, outeredge, inner, outer). . . , 5.2 5.3.
- , . , left right.
- here . . . force : . . . , .
- : „page” ; „opposite” ; „leftpage” ; „rightpage” an odd page.  
. . :
- none: .
- split: . , , . , .

### 13.4.2

- ```
\placefloat „none”, :
```
- . ; , „sekvenca” „sekvenca” , „Sekvenca 1”.
. , „figure” „table” ; „figure” „”, „table” „”. \setuplabeltext 10.5.3.
 - . , 3 '3.1'.
 - . \placefloat.
- ```
\setupcaptions \setupcaption[] . , :
```
- , number, way, prefixsegments numberconversion:
    - number yes, no none .
    - way (way=bytext), (way=bychapter) (way=bysection). , prefixsegments.
    - prefixsegments , . prefixsegments=chapter , prefixsegments=section .
    - numberconversion . : („numbers”), („a”, „characters”), („A”, „Characters”), („KA”), („I”, „R”, „Romannumerals”), („i”, „r”, „romannumerals”) („KR”).
  - . , 7.4.4 , . :
    - style, color, command.
    - : headstyle, headcolor, headcommand, headseparator.
    - : numbercommand.
    - : textcommand.
  - , , , ConTeXt .

### 13.4.3

```
, ConTeXt , \startcombination , :
\startcombination[] ... \stopcombination
: , . , , * :
\startcombination[3*2]
{\externalfigure[test1]}
{\externalfigure[test2]}
{\externalfigure[test3]}
{\externalfigure[test4]}
{\externalfigure[test5]}
{\externalfigure[test6]}
\stopcombination
```

|                                                     |                                                     |                                                     |
|-----------------------------------------------------|-----------------------------------------------------|-----------------------------------------------------|
| <pre> name: test1 file: test1 state: unknown </pre> | <pre> name: test3 file: test3 state: unknown </pre> | <pre> name: test5 file: test5 state: unknown </pre> |
| <pre> name: test2 file: test2 state: unknown </pre> | <pre> name: test4 file: test4 state: unknown </pre> | <pre> name: test6 file: test6 state: unknown </pre> |

```

, ConTEXt .
, \startcombination, .
, \startcombination , startframedtext , . \setupcombination, \definecombination .

```

### 13.4.4

```

\placefloat , . :
• . \setupfloat[].
• . \setupfloats.
\placefloat[figure] \placefigure, \setupfloat[figure] \setupfigures, \setupfloat[table]
\setuptables.
, ConTEXt (3.6).

```

## 13.5

```

\definefloat . :
\definefloat[] [] []
. \setupfloat[].
, ConTEXt , , , .
ConTEXt. :
• .
• . , .
• .
• , , (TEX) .
, : , . ; , , .
ConTEXt: (\setupfloats) : \placefloat: ConTEXt :
1. . \definefloat, , .
2. , , : \setupfloat[].
3. , (\placefloat), : (\placefloat []).
4. , , . () , .
, ConTEXt , ConTEXt:
• .
• , .
• „title” , .
• , . .
13.3 , :

```

---

|                                      |                              |                                     |
|--------------------------------------|------------------------------|-------------------------------------|
| <code>\completelistof&lt;&gt;</code> | <code>\completelist[]</code> | <code>\completelistoffigures</code> |
| <code>\place&lt;&gt;</code>          | <code>\placefloat[]</code>   | <code>\placefigure</code>           |
| <code>\placelistof&lt;&gt;</code>    | <code>\placelist[]</code>    | <code>\placelistoffigures</code>    |
| <code>\setup&lt;&gt;</code>          | <code>\setupfloat[]</code>   | <code>\setupfigure</code>           |

---

13.3

```
, 13.3: \start<>, \start<>text \startplace<>.

; , ConTEXt, \definefloat:

\definefloat[chemical][chemicals]
\definefloat[figure][figures]
\definefloat[table][tables]
\definefloat[intermezzo][intermezzi]
\definefloat[graphic][graphics]

, ConTEXt ; . \placefigure \placetable ., \placefigure , \placetable, .
```





# , ConT<sub>E</sub>Xt

T<sub>E</sub>X (TeX Live, teTeX, MikTeX, MacTeX, ...) ConT<sub>E</sub>Xt. , . ConT<sub>E</sub>Xt; ConT<sub>E</sub>Xt Mark II Mark IV, ConT<sub>E</sub>Xt Mark IV.

; , :

- **Unix** : Unix, GNU Linux, Mac OS, FreeBSD, OpenBSD Solaris. ; .
- **Windows** : Windows 10 ( ), Windows 8, Windows 7, Windows Vista, Windows XP, Windows NT, .

**Microsoft Windows** :

ConT<sub>E</sub>Xt, T<sub>E</sub>X , ; . Windows , , Windows ( ) , , , Windows : „DOS ”, „ ”, „cmd”, . Windows Windows . Windows 2004. , . ; .

## 1 „ConT<sub>E</sub>Xt Standalone”

ConT<sub>E</sub>Xt „Standalone”, „ConT<sub>E</sub>Xt Suite”, ConT<sub>E</sub>Xt, , — *Standalone ()* — ConT<sub>E</sub>Xt , . , ConT<sub>E</sub>Xt Mark II ( T<sub>E</sub>X, pdfT<sub>E</sub>X X<sub>Ǝ</sub>T<sub>E</sub>X ), ConT<sub>E</sub>Xt Mark IV ( LuaT<sub>E</sub>X ).

T<sub>E</sub>X , 1.4.1; T<sub>E</sub>X ConT<sub>E</sub>Xt, Mark II Mark IV, 1.5.1.

, , „ConT<sub>E</sub>Xt Standalone”. ConT<sub>E</sub>Xt , ConT<sub>E</sub>Xt [wikibooks](#). , , ( ).

### 1.1

„ConT<sub>E</sub>Xt Standalone” , :

1. ConT<sub>E</sub>Xt.
2. .
3. .
4. .

#### 1:

, ConT<sub>E</sub>Xt . Windows . Unix , . , Unix , „~/context/” Windows, „C:\Programs\context”.

#### 2:

:

- Unix , „wget” „rsync”:

```
wget http://minimals.contextgarden.net/setup/first-setup.sh
rsync rsync://minimals.contextgarden.net/setup/first-setup.sh
```

- Windows , , . :

<http://minimals.contextgarden.net/setup/context-setup-mswin.zip>  
<http://minimals.contextgarden.net/setup/context-setup-win64.zip>

, Windows .

#### 3:

. Unix , „first-setup.sh” bash sh. Windows , „first-setup.bat” MS-DOS .

```

:
• --context: ConTeXt , („-context=latest”) („-context=beta”). „beta”.
• --engine: Mark IV („-engine=luatex”,) Mark II.
• --modules: ConTeXt , . , „-modules=all”.
, . Mark IV, „-engine=luatex”, „-context=latest” , ., 2020. , first-setup.sh , „-context=latest”,
„-engine=luatex” .

(ConTeXt): „-fonts=all” „goodies=all”. ConTeXtgarden , . (Unix Windows):
• Unix: bash first-setup.sh --context=latest --modules=all --fonts=all --goodies=all
• Windows: first-setup.bat --context=latest --modules=all --fonts=all --goodies=all

, , .

rsync . , , rsync . RSYNC_PROXY, (.bashrc). , . Unix „export”, Windows „set”. :
export RSYNC_PROXY=:0:
, , 873 TCP . 22 ssh , 873 (nc).
export RSYNC_CONNECT_PROG='ssh nc %H 873'
” , , nc 873 TCP .

„first-setup” , „bin” „tex”.

```

#### 4: ( GNU Linux)

GNU Linux . ConTeXt , . , „tex/setuptex” :

```

export OSFONTDIR="/usr/share/fonts:/usr/share/texmf/fonts/opentype/"

OSFONTDIR .

/usr/share/texmf/fonts/ TEX ; OSFONTDIR opentype . ConTeXt , OSFONTDIR , . , , /usr/local/fonts
/usr/share/fonts.

, ConTeXt . :

. ~/context/tex/setuptex
context --generate
context --make

. bash source. , source.

```

### 1.2 „ConTeXt Standalone”

„ConTeXt Standalone” T<sub>E</sub>X , ; , ConTeXt , „context”, „tex/setuptex” (Unix) „tex/setuptex.bat” (Windows). :

- Unix , „context”, :

```

source ~/context/tex/setuptex
. ~/context/tex/setuptex

(„context” „~/context”).

```
  - Windows , **tex\setuptex.bat** ConTeXt.

```

TEX , :

```
  - Unix ( „.bashrc”).

```

. bash (GNU Linux), .bashrc.sh ksh .profile, zsh .zshenv, tcsh csh .cshrc . , .bashrc .bash_profile.

```
  - Windows cmd.exe , :

```

C:\WINDOWS\System32\cmd.exe /k C:\Programs\context\tex\setuptex.bat

```
- , ConTeXt, , , ConTeXt . , . ConTeXt : LEd, Notepad++, Scite, TeXnicCenter, TeXworks, vim .

### 1.3 „ConTeXt Standalone”

Mark IV, „ConTeXt Standalone” . , : „first-setup.sh” .  
 „ConTeXt Standalone”, „first-setup” „--context=”, . --.  
 ConTeXt .  
 , , , GNU Linux 4 , „”.

## 2 LMTX

ConTeXt Mark IV, LuaTeX LuaMetaTeX, LuaTeX , „ConTeXt Standalone” LMTX.  
 ConTeXt. TEX : LuaMetaTeX. 2019. 2020. ConTeXt [ConTeXt](#) .  
 LMTX , , Mark IV, , LMTX (4 2020) \Caps . „ConTeXt Standalone”.

### 2.1

:

- **1:** LMTX, , „context” .
- **2:** ( ) zip [ConTeXt](#) . :
  - GNU/Linux
    - ★ X86
      - ▷ 32 .
      - ▷ 64 .
    - ★ ARM
      - ▷ 32 .
      - ▷ 64 .
  - Microsoft Windows
    - ★ 32
    - ★ 64
  - Mac OS, 64
  - FreeBSD
    - ★ 32 .
    - ★ 64 .
  - OpenBSD 6.6
    - ★ 32 .
    - ★ 64 .
  - OpenBSD 6.7
    - ★ 32 .
    - ★ 64 .
- 32- 64-, – – 64-. X86 ARM, X86.
- **3:** . „bin” , „installation.pdf”, , „install.sh” ( Unix ) „install.bat” ( Windows ).
- **Step 4:** („install.sh” „install.bat”). , .
  - Unix , bash, sh. , „home” () .
  - Windows , install.bat. , , .
- **5** LMTX:
 

Windows , „setpath.bat” Windows LMTX . GNU Linux , FreeBSD Mac OS ,  
 ConTeXt PATH , :

```
export PATH="/tex/texmf-/bin:"$PATH
```

( , „/home/user/context”) texmf- LMTX . , 64 Linux , texmf- „texmf-linux-64”, :

```
export PATH="/home/user/context/texmf-linux-64/bin:"$PATH
```

LMTX , , : bash, sh, zsh, ksh, tcsh, csh... Linux , , „.bashrc”, :

```
echo 'export PATH="/home/user/context/texmf-linux-64/bin:"$PATH >> .bashrc
```

: , ConTeXt , TeX Live „ConTeXt Standalone”. , 3.

## 2.2 LMTX

ConTeXt LMTX does ConTeXt . , ConTeXt .  
 , LMTX ( install.sh install.bat) . GNU Linux . Windows , .

## 2.3 LMTX

LMTX : , .  
 , ; „install.sh” „install.bat”; , .

## 2.4 LMTX ( GNU/Linux )

, „ConTeXt Standalone” („tex/setuptex”) , LMTX . , , „setuplmtx” „tex”. :

```
export PATH="/context/LMTX/tex/texmf-linux-64/bin:$PATH"
echo " ~/context/LMTX/tex/texmf-linux-64/bin PATH"
export TEXROOT="/context/LMTX/tex"
echo " ~/context/LMTX/tex TEXROOT"
export OSFONTDIR="/.fonts:/usr/share/fonts:/usr/share/texmf/fonts/opentype/"
echo " "
alias lmtx="/context/LMTX/tex/texmf-linux-64/bin/context"
echo " lmtx"
```

, LMTX, „lmtx” „context”.

, LMTX, , :

```
source ~/context/LMTX/tex/setuplmtx
```

```
LMTX „~/context/LMTX” „setuplmtx” „~/context/LMTX/tex”.
```

```
LMTX „ConTeXt Standalone”. , LMTX , , OSFONTDIR, ConTeXt .
```

## 3 ConTeXt ( Unix )

```
alias ConTeXt . , ConTeXt TeX Live LMTX; Standalone LMTX.
```

, LMTX 2020. , „.bashrc” ( ):

```
alias lmtx-01="/home/user/context/202001/tex/texmf-linux-64/bin/context"
alias lmtx-08="/home/user/context/202008/tex/texmf-linux-64/bin/context"
```

```
lmtx-01 LMTX „context/202001”, lmtx-08 „context/202008”.
```

, ; ( ) .  
 . , ; . , , ( , , , , ) . , ( ) .  
 , . ConTeXt , . TeX LaTeX, ConTeXt. LaTeX ( '\$'). ConTeXt, , .

|   |               |   |               |    |             |
|---|---------------|---|---------------|----|-------------|
| © | \copyright    | ® | \registered   | ¢  | \textcent   |
| ⊙ | \textcircledP | ⌘ | \textcurrency | \$ | \textdollar |
| Ⓓ | \textdong     | € | \texteuro     | f  | \textflorin |
| £ | \textsterling | ¥ | \textyen      | ™  | \trademark  |

, ,

|   |                     |   |               |   |              |
|---|---------------------|---|---------------|---|--------------|
| △ | \triangle           | ○ | \bigcirc      | □ | \square      |
| ◁ | \triangleleft       | ◦ | \circ         | ■ | \blacksquare |
| ▷ | \triangleright      | • | \bullet       | ◻ | \boxdot      |
| ▽ | \triangledown       | ⊗ | \circledast   | ◻ | \boxminus    |
| ▼ | \blacktriangledown  | ⊙ | \circledcirc  | ⊞ | \boxplus     |
| ◄ | \blacktriangleleft  | ⊖ | \circledminus | ⊠ | \boxtimes    |
| ► | \blacktriangleright | ⊕ | \bigoplus     | * | \ast         |
| ▲ | \blacktriangle      | ⊗ | \bigotimes    | ✠ | \maltese     |
| ≐ | \triangleq          | ⊕ | \oplus        | * | \star        |
| ◇ | \diamond            | ⊖ | \ominus       | ♣ | \clubsuit    |
| ◊ | \lozenge            | ⊗ | \otimes       | ♥ | \heartsuit   |
| ◆ | \blacklozenge       | ⌘ | \oslash       | ♠ | \spadesuit   |
| ◇ | \diamondsuit        | ○ | \odot         | ∅ | \varnothing  |

:

|   |                   |   |                            |    |                      |
|---|-------------------|---|----------------------------|----|----------------------|
| ← | \leftarrow, \gets | → | \rightarrow, \to           | ↔  | \leftrightarrow      |
| ↔ | \nleftarrow       | ↗ | \nrightarrow               | ↔  | \Leftrightarrow      |
| ↔ | \longleftarrow    | ↗ | \longrightarrow            | ↔  | \longleftrightarrow  |
| ↔ | \Lleftarrow       | ↗ | \Rrightarrow               | ↔  | \Longleftrightarrow  |
| ↔ | \nLeftarrow       | ↗ | \nRrightarrow              | ↔  | \leftrightharpoons   |
| ↔ | \Lsh              | ↗ | \Rsh                       | ↔  | \leftrightsquigarrow |
| ↔ | \mapsfrom         | ↗ | \mapsto                    | ↔  | \nleftrightarrow     |
| ↔ | \longmapsfrom     | ↗ | \longmapsto                | ↔  | \nletrightarrow      |
| ↔ | \Mapsfrom         | ↗ | \Mapsto                    | ↔  | \rightleftarrows     |
| ↔ | \Longmapsfrom     | ↗ | \Longmapsto                | ↔  | \rightleftharpoons   |
| ↔ | \leftarrowtail    | ↗ | \rightarrowtail            | ↕  | \updownarrow         |
| ↔ | \twoheadleftarrow | ↗ | \twoheadrightarrow         | ↕  | \Updownarrow         |
| ↔ | \circlearrowleft  | ↗ | \circlearrowright          | ↕  | \updownarrows        |
| ↔ | \curvearrowleft   | ↗ | \curvearrowright           | ↑  | \uparrow             |
| ↔ | \hookleftarrow    | ↗ | \hookrightarrow            | ↑  | \Uparrow             |
| ↔ | \leftharpoonup    | ↗ | \rightharpoonup            | ↑↑ | \upuparrows          |
| ↔ | \leftharpoonup    | ↗ | \rightharpoonup            | ↑  | \twoheaduparrow      |
| ↔ | \leftleftarrows   | ↗ | \rightrightarrows          | ↑  | \upharpoonleft       |
| ↔ | \looparrowleft    | ↗ | \looparrowright            | ↑  | \upharpoonright      |
| ↔ | \swarrow          | ↗ | \searrow                   | ↓  | \downarrow           |
| ↔ | \nwarrow          | ↗ | \nearrow                   | ↓  | \Downarrow           |
| ↔ | \leftsquigarrow   | ↗ | \leadsto, \rightsquigarrow | ↓↓ | \downdownarrows      |
| ↔ | \\$iff\$          | ↓ | \twoheaddownarrow          | ↓  | \downharpoonleft     |
| ⇒ | \\$implies\$      |   |                            | ↓  | \downharpoonright    |

|   |          |   |            |   |        |
|---|----------|---|------------|---|--------|
| ∵ | \because | ⋅ | \cdot      | ⋅ | \cdot  |
| ⋮ | \cdots   | ⋅ | \centerdot | : | \colon |
| ⋮ | \ddots   | ⋮ | \dots      | ⋅ | \ldotp |

... \ldots    ... \textellipsis    ∴ \therefore  
 ∷ \vdots    „ \quotedblbase    " \quotedbl

|    |                 |   |                   |
|----|-----------------|---|-------------------|
| :  |                 |   |                   |
| ℵ  | \aleph          | ℙ | \amalg            |
| ≈  | \approx         | ≈ | \approxeq         |
| ↖  | \backsim        | \ | \backslash        |
| ∅  | \between        | ⊃ | \bigcap           |
| ⊔  | \bigsqcup       | ⊕ | \biguplus         |
| ∧  | \bigwedge       | ⊥ | \bot              |
| ⊙  | \bumpeq         | ⊂ | \cap              |
| ⊖  | \circeq         | ⊂ | \complement       |
| ⊕  | \coprod         | ⊂ | \cup              |
| ⋈  | \curlyeqprec    | ⋈ | \curlyeqsucc      |
| ⋈  | \curlywedge     | ⋈ | \dashv            |
| ‡  | \ddagger, \ddag | ⋈ | \diamondsuit      |
| ⋈  | \divideontimes  | ⋈ | \doteq            |
| +  | \dotplus        | ℓ | \ell              |
| #  | \eqcirc         | ≃ | \eqslantgtr       |
| ≡  | \equiv          | ⊂ | \eth              |
| ∃! | \exists!        | ≡ | \fallingdotseq    |
| ∀  | \forall         | ⋈ | \frown            |
| ∇  | \geqslant       | ⋈ | \gg               |
| ∇  | \gnapprox       | ≃ | \gneq             |
| ∇  | \gtapprox       | ∇ | \gtrdot           |
| ∇  | \gtreqless      | ≈ | \gtrless          |
| h  | \hbar           | ♥ | \heartsuit        |
| ⋈  | \iiint          | ℐ | \Im               |
| ∈  | \in             | ∞ | \infty            |
| ⊔  | \intercal       | ⋈ | \jmath            |
| ⋈  | \leftthreetimes | ⋈ | \leq, \le         |
| ⋈  | \leqslant       | ⋈ | \lessapprox       |
| ⋈  | \lesseqgtr      | ⋈ | \lesseqgtr        |
| ⋈  | \lessim         | ⋈ | \ll               |
| ⋈  | \lnapprox       | ⋈ | \lneq             |
| ⋈  | \lnsim          | ⋈ | \lor              |
| ⋈  | \measuredangle  | ⋈ | \models           |
| ⋈  | \multimap       | ⋈ | \nVDash           |
| ⋈  | \natural        | ⋈ | \ncong            |
| ⋈  | \neg o \lnot    | ⋈ | \nexists          |
| ⋈  | \ngtr           | ⋈ | \ni               |
| ⋈  | \nless          | ⋈ | \nmid             |
| ⋈  | \not\equiv      | ⋈ | \not\sim          |
| ⋈  | \notin          | ⋈ | \nparallel        |
| ⋈  | \nsim           | ⋈ | \nsubseteq        |
| ⋈  | \nsubseteq      | ⋈ | \ntriangleleft    |
| ⋈  | \ntriangleright | ⋈ | \ntrianglerighteq |
| ⋈  | \nvDash         | ⋈ | \oint             |
| ⋈  | \partial        | ⋈ | \perp             |
| ⋈  | \pm             | ⋈ | \prec             |
| ⋈  | \preceq         | ⋈ | \precnsim         |
| ⋈  | \prime          | ⋈ | \prod             |
| ⋈  | \Re             | ⋈ | \rightthreetimes  |
| ⋈  | \rtimes         | ⋈ | \sharp            |
| ⋈  | \simeq          | ⋈ | \smile            |
| ⋈  | \sqcap          | ⋈ | \sqcup            |
| ⋈  | \sqsubseteq     | ⋈ | \sqsupseteq       |
| ⋈  | \subset         | ⋈ | \Subset           |
| ⋈  | \subsetneq      | ⋈ | \succ             |
| ⋈  | \succeq         | ⋈ | \succsim          |
| ⋈  | \sum            | ⋈ | \supseteq         |
| ⋈  | \supseteq       | ⋈ | \supsetneq        |
| ⋈  | \textpm         | ⋈ | \times            |
| ⋈  | \triangle       | ⋈ | \uplus            |
| ⋈  | \Vdash          | ⋈ | \vee o \lor       |
| ⋈  | \Vert           | ⋈ | \Vdash            |
| ⋈  | \wp             | ⋈ | \wr               |

|   |             |    |             |    |                   |
|---|-------------|----|-------------|----|-------------------|
| ° | \P          | §  | \S          | °C | \celsius          |
| ✓ | \checkmark  | Ω  | \mho        | Ω  | \ohm              |
| ° | \textdegree | N° | \textnumero | □  | \textvisiblespace |





`\ccedilla` 107  
`\cf` 82  
`\chapter` 87  
`\chi` 108  
`\clip` 139  
`\clubpenalty` 123  
`\color` 85  
`\colored` 85  
`\completecontent` 93  
`\compleitelist` 97  
`\completelistofchemicals` 97  
`\completelistoffigures` 97, 138  
`\completelistofgraphics` 97  
`\completelistofintermezzi` 97  
`\completelistoftables` 97, 139  
`\completindex` 99  
`\crlf` 120  
`\currentdate` 115

**d**

`\date` 115  
`\de` 114  
`\define` 42  
`\definealternativestyle` 83  
`\defineblank` 118  
`\definebodyfontenvironment` 82  
`\definebodyfontswitch` 83  
`\definecapitals` 111  
`\definecharacter` 108  
`\definecharacterkerning` 112  
`\definecolor` 86  
`\definecombination` 143  
`\definecombinedlist` 97  
`\defineconversionset` 71  
`\definedelimitedtext` 116  
`\definedescription` 131  
`\defineenumeration` 132  
`\definefloat` 143  
`\definefontfeature` 108  
`\definefontstyle` 83  
`\defineframed` 134  
`\defineframedtext` 134  
`\definehead` 91  
`\defineinterlinespace` 121  
`\defineitemgroup` 131  
`\defineitems` 131  
`\definelayou` 69  
`\definelinenumbering` 122  
`\definelines` 121  
`\definelist` 97  
`\definemargindata` 77  
`\definennarrower` 118  
`\definernote` 125  
`\definepapersize` 64  
`\defineparagraphs` 128  
`\defineregister` 99  
`\definesectionblock` 92  
`\definestartstop` 44  
`\definestretched` 112  
`\definesymbol` 129  
`\definetext` 75  
`\definetype` 112  
`\definotyping` 112  
`\delta` 108  
`\dontleavehmode` 81

**e**

`\eacute` 107  
`\ebreve` 107  
`\ecircumflex` 107  
`\ediaeresis` 107  
`\egrave` 107  
`\egroup` 45, 46  
`\em` 84  
`\emacron` 107  
`\emdash` 54  
`\en` 114  
`\enableregime` 51  
`\endash` 54  
`\endgraf` 117  
`\endgroup` 46  
`\endnote` 124  
`\enskip` 113  
`\environment` 57  
`\epsilon` 108  
`\es` 114  
`\eta` 108  
`\etilde` 107  
`\externalfigure` 136

**f**

`\fillinline` 133  
`\footnote` 124  
`\fr` 114  
`\framed` 134  
`\from` 104

**H**

`\H` 107

**g**

`\gamma` 108  
`\getbuffer` 134  
`\getmarking` 74

**H**

`\HL` 140

**g**

`\godown` 119  
`\goto` 104

**h**

`\hairline` 133  
`\handwritten` 82  
`\hbox` 120  
`\head` 130  
`\hfill` 113  
`\high` 111  
`\hl` 133  
`\hskip` 113  
`\hw` 82  
`\hyphen` 54  
`\hyphenatedurl` 104  
`\hyphenatedurlseparator` 104  
`\hyphenation` 120

**i**

`\i` 107  
`\iacute` 107  
`\ibreve` 107

`\icircumflex` 107  
`\idiaeresis` 107  
`\igrave` 107  
`\imacron` 107  
`\in` 101  
`\index` 98  
`\inframed` 134  
`\ininner` 75  
`\ininneredge` 75  
`\innermargin` 75  
`\inleft` 75  
`\inleftedge` 75  
`\inleftmargin` 75  
`\inmargin` 75  
`\inothet` 75  
`\inouter` 75  
`\inouteredge` 75  
`\inoutermargin` 75  
`\input` 55  
`\inright` 75  
`\inrightedge` 75  
`\inrightmargin` 75  
`\iota` 108  
`\it` 82  
`\ita` 82  
`\italic` 82  
`\italicbold` 82  
`\itb` 82  
`\itc` 82  
`\itd` 82  
`\item` 129  
`\items` 131  
`\itilde` 107  
`\its` 130  
`\itx` 82  
`\itxx` 82

**j**  
`\j` 107

**k**  
`\kappa` 108  
`\kcedilla` 107

**l**  
`\l` 107  
`\labeltext` 115  
`\lambda` 108  
`\language` 113  
`\lastpagenumber` 72  
`\lastrealpagenumber` 72  
`\lastuserpagenumber` 72  
`\lcedilla` 107  
`\leftaligned` 123  
`\letterbackslash` 104  
`\letterescape` 104  
`\letterhash` 104  
`\letterhat` 34  
`\letterpercent` 104  
`\lettertilde` 34  
`\linenote` 124  
`\loadinstalledlanguages` 113  
`\lohi` 111  
`\low` 111

**N**  
`\NB` 140  
  
**m**  
`\mainlanguage` 113  
`\mar` 130  
`\margintext` 75

**N**  
`\NC` 140

**m**  
`\mediaeval` 82  
`\midaligned` 123  
`\minus` 54  
`\mirror` 139

**N**  
`\NN` 140

**m**  
`\mono` 82  
`\month` 115

**N**  
`\NR` 140

**m**  
`\mu` 108

**n**  
`\ncedilla` 107  
`\noheaderandfooterlines` 74  
`\noindentation` 117  
`\nolist` 88, 89  
`\nomarking` 88, 89  
`\normal` 82  
`\note` 125  
`\notesenabledfalse` 127  
`\notesenabledtrue` 127  
`\nowhitespace` 118  
`\nu` 108

**o**  
`\o` 107  
`\oacute` 107  
`\obreve` 107  
`\ocircumflex` 107  
`\odiaeresis` 107  
`\oe` 108  
`\oeligature` 108  
`\ograve` 107  
`\omacron` 107  
`\omega` 108  
`\omicron` 108  
`\os` 82  
`\otilde` 107  
`\overbar` 134  
`\overbars` 134  
`\overstrike` 134  
`\overstrikes` 134

**p**  
`\page` 72

`\pagenumber` 71, 74  
`\pagereference` 101  
`\par` 117  
`\parindent` 48  
`\parskip` 48  
`\part` 87  
`\phi` 108  
`\pi` 108  
`\placebookmarks` 105  
`\placechemical` 141  
`\placecontent` 93  
`\placefigure` 137  
`\placefloat` 141  
`\placegraphic` 141  
`\placeindex` 99  
`\placeintermezzo` 141  
`\placelist` 97  
`\placelistofchemicals` 97  
`\placelistoffigures` 97, 138  
`\placelistofgraphics` 97  
`\placelistofintermezzi` 97  
`\placelistoftables` 97, 139  
`\placelocalfootnotes` 125  
`\placenotes` 125  
`\placetable` 139  
`\pretolerance` 120  
`\product` 58  
`\project` 59, 60  
`\psi` 108

**R**

`\ReadFile` 57

**q**

`\qqquad` 113  
`\quad` 113  
`\quotation` 116  
`\quote` 116

**r**

`\r` 107  
`\ran` 130  
`\rcedilla` 107  
`\readfile` 57  
`\realpagenumber` 71  
`\ref` 102  
`\reference` 101  
`\regular` 82  
`\rho` 108  
`\rightaligned` 123  
`\rm` 82  
`\rma` 82  
`\rmb` 82  
`\rmc` 82  
`\rmd` 82  
`\rmx` 82  
`\rmxx` 82  
`\roman` 82  
`\rotate` 139

**s**

`\sans` 82  
`\sansserif` 82  
`\sc` 82  
`\scedilla` 107

`\section` 87  
`\seeindex` 99  
`\serif` 82  
`\sethyphenatedurlafter` 104  
`\sethyphenatedurlbefore` 104  
`\sethyphenatedurlnormal` 104  
`\setupalign` 122  
`\setuparranging` 68  
`\setupbackgrounds` 84  
`\setupblank` 118  
`\setupbodyfont` 80, 81  
`\setupbottomtexts` 75  
`\setupcapitals` 111  
`\setupcaption` 142  
`\setupcaptions` 142  
`\setupcharacterkerning` 112  
`\setupcolors` 84  
`\setupcolumns` 127  
`\setupcombinedlist` 94  
`\setupcounter` 125  
`\setupdescription` 132  
`\setupendnotes` 126  
`\setupenumeration` 133  
`\setuexternalfigures` 138  
`\setupfillinlines` 133  
`\setupfloat` 143  
`\setupfloats` 143  
`\setupfooter` 74  
`\setupfootertexts` 73, 75  
`\setupfootnotes` 126  
`\setupframed` 134  
`\setupframedtext` 134  
`\setuphead` 88  
`\setupheader` 74  
`\setupheadertexts` 73, 75  
`\setupheadnumber` 89  
`\setupheads` 88  
`\setupheadtext` 93  
`\setuphyphenmark` 113  
`\setupindenting` 117  
`\setupinteraction` 103  
`\setupinterlinespace` 121  
`\setuplabeltext` 115  
`\setuplanguage` 114  
`\setuplayout` 66  
`\setuplinenote` 124  
`\setuplinenumbers` 122  
`\setuplines` 121  
`\setuplist` 95  
`\setupmargindata` 76  
`\setupnarrower` 118  
`\setupnotation` 125  
`\setupnotations` 126  
`\setupnote` 125  
`\setupnotes` 126  
`\setuppagenumbering` 70  
`\setuppapersize` 62  
`\setupparagraphs` 128  
`\setupregister` 99  
`\setupsectionblock` 92  
`\setupspacing` 112  
`\setupstretched` 112  
`\setuptables` 139  
`\setuptextrule` 134  
`\setuptolerance` 120, 123

`\setuptoptexts` 75  
`\setuptype` 112  
`\setuptyping` 112  
`\setupurl` 104  
`\setupuserpagenumber` 70  
`\setupwhitespace` 118  
`\showbodyfontenvironment` 83  
`\showcolor` 85, 86  
`\showcolorcomponents` 86  
`\showfont` 81  
`\showframe` 66  
`\showinstalledlanguages` 113  
`\showlayout` 66  
`\showsetups` 66  
`\showsymbolset` 110  
`\sigma` 108  
`\sl` 82  
`\sla` 82  
`\slanted` 82  
`\slantedbold` 82  
`\slb` 82  
`\slc` 82  
`\sld` 82  
`\slx` 82  
`\slxx` 82  
`\smalcaps` 82  
`\smallbodyfont` 83  
`\smallbold` 83  
`\smallbolditalic` 83  
`\smallboldslanted` 83  
`\smallitalicbold` 83  
`\smallslanted` 83  
`\smallslantedbold` 83  
`\somewhere` 102  
`\space` 113  
`\ss` 82, 107  
`\ssa` 82  
`\ssb` 82  
`\ssc` 82  
`\ssd` 82  
`\ssx` 82  
`\ssxx` 82  
`\start` 46  
`\startTEXpage` 64  
`\startalignment` 123  
`\startappendices` 92  
`\startbackmatter` 92  
`\startbodymatter` 92  
`\startbuffer` 134  
`\startchapter` 87  
`\startchemical` 134  
`\startcolor` 85  
`\startcolumns` 127  
`\startcombination` 134, 142  
`\startcomponent` 58  
`\startenvironment` 57  
`\startfiguretext` 138  
`\startformula` 134  
`\startframedtext` 134  
`\startfrontmatter` 92  
`\starthiding` 134  
`\startitem` 129  
`\startitemize` 129  
`\startlegend` 134  
`\startlinecorrection` 134  
`\startlinenumbers` 122  
`\startlines` 121  
`\startlocalfootnotes` 125  
`\startMPpage` 64  
`\startmode` 134  
`\startnarrower` 117  
`\startnotmode` 134  
`\startopposite` 134  
`\startpacked` 118  
`\startpagefigure` 64  
`\startpart` 87  
`\startproduct` 58  
`\startproject` 59  
`\startquotation` 135  
`\startsection` 87  
`\startsetups` 45  
`\startstandardmakeup` 135  
`\startsubject` 87  
`\startsubsection` 87  
`\startsubsubsection` 87  
`\startsubsubsubject` 87  
`\startsubsubsubsection` 87  
`\startsubsubsubsubject` 87  
`\starttabulate` 139  
`\starttext` 54  
`\starttextrule` 134  
`\starttitle` 87  
`\starttyping` 111  
`\stop` 46  
`\stoptext` 54  
`\stretched` 112  
`\structureuservariable` 88  
`\sub` 130  
`\subject` 87  
`\subsection` 87  
`\subsubject` 87  
`\subsubsection` 87  
`\subsubsubject` 87  
`\subsubsubsection` 87  
`\subsubsubsubject` 87  
`\support` 82  
`\switchtobodyfont` 81  
`\sym` 130  
`\symbol` 110  
  
**t**  
`\tau` 108  
`\tcedilla` 107  
`\teletype` 82  
`\tex` 112  
`\textheight` 68  
`\textreference` 101  
`\textrule` 134  
`\l` 113  
`\textwidth` 68  
`\tf` 82  
`\tfa` 82  
`\tfb` 82  
`\tfc` 82  
`\tfd` 82  
`\tfx` 82  
`\tfxx` 82  
`\theta` 108  
`\thinrule` 133  
`\thinrules` 133

`\title` 87  
`\tolerance` 120  
`\translate` 115  
`\tt` 82  
`\tta` 82  
`\ttb` 82  
`\ttc` 82  
`\ttd` 82  
`\ttx` 82  
`\ttxx` 82  
`\tx` 82  
`\txx` 82  
`\typ` 112  
`\type` 111

**u**

`\u` 107  
`\uacute` 107  
`\ubreve` 107  
`\ucircumflex` 107  
`\udiaeresis` 107  
`\ugrave` 107

**V**

`\VL` 140

**u**

`\umacron` 107  
`\underbar` 134  
`\underbars` 134  
`\unhyphenated` 120  
`\upsilon` 108  
`\usecolors` 85  
`\useexternalfigure` 136  
`\usemodule` 113  
`\useregime` 51  
`\userpagenumber` 71  
`\usesymbols` 110

`\useURL` 103  
`\utilde` 107

**v**

`\varepsilon` 108  
`\varkappa` 108  
`\varphi` 108  
`\varpi` 108  
`\varrho` 108  
`\varsigma` 108  
`\vartheta` 108  
`\vbox` 73  
`\vfill` 119  
`\vl` 133

**W**

`\WORD` 111  
`\WORDS` 111  
`\Word` 111  
`\Words` 111

**v**

`\vskip` 119

**w**

`\whitespace` 118  
`\widowpenalty` 123  
`\word` 111  
`\wordright` 123  
`\writebetweenlist` 96  
`\writetolist` 96

**x**

`\xi` 108

**z**

`\zeta` 108